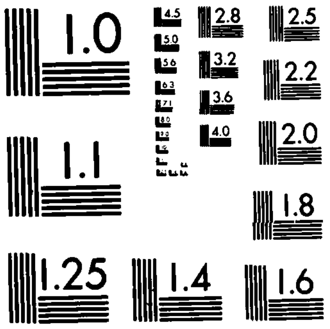
AD-A123 632 SUPERVISORY CONTROL OF UNDERWATER TELEMANIPULATORS: DESIGN AND EXPERIMENT(U) MASSACHUSETTS INST OF TECH CAMBRIDGE MAN-MACHINE SYSTEMS LAB D R YOERGER 1/3

UNCLASSIFIED 30 AUG 82 N00014-77-C-0256 F/G 5/8 NL



MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963-A

SUPERVISORY CONTROL OF UNDERWATER TELEMANIPULATORS: DESIGN AND EXPERIMENT

Dana R. Yoerger

August 1982

CONTRACT N00014-77-C-0256
WORK UNIT NR 196-158
ENGINEERING PSYCHOLOGY PROGRAMS
OFFICE OF NAVAL RESEARCH
ARLINGTON, VIRGINIA 22217

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE

READ INSTRUCTIONS BEFORE COMPLETING FORM

1. REPORT NUMBER	
2. GOVT ACCESSION NO.	AD-A123632
3. RECIPIENT'S CATALOG NUMBER	
4. TITLE (and Subtitle)	Supervisory Control of Underwater Telemanipulators: Design and Experiment
5. TYPE OF REPORT & PERIOD COVERED	Technical Report June 1,1979 - Sept.30,1982
6. PERFORMING ORG. REPORT NUMBER	
7. AUTHOR(s)	Dana R. Yoerger
8. CONTRACT OR GRANT NUMBER(s)	N00014-77-C-0256
9. PERFORMING ORGANIZATION NAME AND ADDRESS	Massachusetts Institute of Technology Cambridge, MA 02139
10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS	NR 196-152
11. CONTROLLING OFFICE NAME AND ADDRESS	Engineering Psychology Group Office of Naval Research, (Code 442EP) Arlington, VA 22217
12. REPORT DATE	Aug. 30, 1982
13. NUMBER OF PAGES	219
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office)	
15. SECURITY CLASS. (of this report)	Unclassified
15a. DECLASSIFICATION/DOWNGRADING SCHEDULE	

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution unlimited.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

None.

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

supervisory control	teleoperators	robotics
man-machine systems	remote control	manual control
human performance	remote undersea vehicles	human factors
man-computer interaction	unmanned vehicles	computer control

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

Supervisory control was investigated as a means to improve the performance of remotely controlled underwater manipulator systems. A system was designed and implemented on a laboratory manipulator and on an underwater device.

The system features an interactive movement control language that allows the human operator to describe motions both analogically and symbolically.

The operator may also interactively combine these movement descriptions to form tasks. The system also includes a computer graphic display that can be used for monitoring the remote manipulator, testing programs, and training operators.

Experiments were run to test the specific skills required of the human operator while teaching the remote system analogically. Performance was quantified and found to be dominated by perceptual effects. Experiments were also run to test features of the movement control language. A combination of analogic and symbolic programming was found to increase accuracy substantially, to decrease the dependence on visual feedback and to decrease variability between subjects in a simulated remote inspection task.

SUPERVISORY CONTROL OF UNDERWATER TELEMANIPULATORS: DESIGN AND EXPERIMENT

by

Dana R. Yoerger

S.B. Massachusetts Institute of Technology
(1977)

S.M. Massachusetts Institute of Technology
(1979)

SUBMITTED TO THE DEPARTMENT OF
MECHANICAL ENGINEERING IN PARTIAL
FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF

DOCTOR OF PHILOSOPHY IN
MECHANICAL ENGINEERING

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

August 1982

Signature of Author...
Department of Mechanical Engineering
August 19, 1982

Certified by............................[illegible]...........
Thomas B. Sheridan
Thesis Supervisor

Accepted by...
Warren M. Rohsenow
Chairman, Departmental Graduate Committee

Acknowledgements

The opportunity to work on this project has allowed me to grow both professionally and personally in ways in which I could not have imagined when I started. In pursuing this work, I travelled across the country and met people from all over the world. All that I have learned could not possibly appear in the pages that follow.

I would like to thank my wife Beverly, and our daughters Lisa and Dora for their patience and help.

I would also like to thank my advisor, Thomas Sheridan, for the opportunities I have had at MIT the last five years. Also, my coworkers Hamy Kazerooni and Ahmet Buharali deserve thanks for helping me and putting up with me. Josef Tzelgov also deserves my thanks for teaching me more about experimentation than perhaps I wanted to know.

I wish to thank the Office of Naval Research, especially Gerald Malecki,for their support of our research efforts and for making this educational opportunity available to me. Thanks also go to Paul Heckman of the Naval Ocean Systems Laboratory for his enthusiasm and support during this project.

Thesis Table of Contents

List of Figures

Chapter 1

INTRODUCTION

1.1 Overview of the thesis

Performing manipulation remotely may be called telemanipulation. In this thesis, supervisory control techniques will be applied to underwater telemanipulation. In particular, capabilities relating to underwater inspection will be examined.

Supervisory control is a paradigm where a person and a computer jointly control a remote system. The computer is under the direction of a human operator. The person decides how the computer should be used to make control of the remote system more effective despite limited communication.

A brief outline of the thesis:

1. First, the need for more effective remotely operated underwater manipulator systems will be discussed.

2. The general concepts of supervisory control will be presented, and then developed for telemanipulation.

3. Techniques for controlling the motions of a manipulator will be discussed, then different types of human oriented interfaces for achieving such control will be presented.

4. The design of a new man-machine interface system intended

specifically for telemanipulation will then be described.

5. A series of experiments that describe the performance of various elements of this new ·stem will then be discussed.

1.2 How Can Supervisory Control Improve Underwater Telemanipulation?

Performing manipulation remotely has always been recognized as a difficult task. However, the opportunities for reducing the risk to human life and the cost of underwater maintenance, inspection, and construction operations have produced much interest in underwater telemanipulation [1], [2], [3].

A variety of experimental studies have been conducted in the U.S. and in countries involved in North Sea oil production. Some of these studies have involved underwater testing of prototype telemanipulation systems [4], [5]. Control of these systems was primarily manual, with some simple computer aiding. While these studies will be discussed in more detail in chapter 5, the conclusions are briefly summarized here:

1. Performance times for a variety of relevent tasks were very long. Much time was spent changing tools and moving cameras.

2. Performance was erratic and often unpredictable.

3. Performance was highly dependent on viewing conditions. Television viewing was the only source of feedback to the operator in the systems tested.

4. Computer aiding seemed to be useful in improving problems

in these first three areas.

The strategy pursued in this thesis will differ from the efforts summarized above. Rather than adding some computerized functions when manual control seems insufficient, concepts relating to the sharing of manipulation tasks between man and computer will be examined. The major goals of this thesis will be:

1. to produce a system that demonstrates a variety of supervisory functions for a practical hardware configuration.

2. to perform experiments that will help designers decide which of these functions are effective and when they are useful.

1.3 What is Supervisory Control?

Supervisory control refers to methods for controlling complex systems in which the system is controlled by a computer which is under the direction of a human operator [6]. Supervisory control fits on a continuum between manually controlled systems and autonomous systems.

In a manually controlled system, control decisions depend on the human operator, although a computer may assist the operator by transforming the inputs from the operator. An example of a computer assisted manual control system is a Resolved Motion Rate Control manipulator system [7] (this type of manipulator control system will be discussed in more detail later). Although the computer assists the human by allowing him to describe velocities in any convenient cartesian coordinate system, the human operator must continuously control the velocity of the arm.

In an autonomous system, a human operator cannot influence the system while it is running. An example of such an autonomous system would be an underwater vehicle which is unable to communicate to human operators, or any form of industrial automation which is designed to operate completely unattended by people.

Systems that lie between manual and autonomous systems are supervisory control systems. The dynamic plant is controlled by a computer system, but the computer system is under the direction of a human operator. The operator can interact with the computer system to adjust setpoints of automatic controllers, to change control algorithms, or to create new algorithms. The human may also use the computer to learn about the state of the dynamic plant and the control system. Generally the operator may request that the system or some portion of the system be placed in a manual control mode.

A wide variety of systems can be classified as supervisory control systems. These include power plants and networks, aircraft, process control systems, and remotely controlled robots and manipulators. All these different systems share a number of important characteristics with regard to how people and machines work together.

1.4 Modelling Supervisory Control

1.4.1 Descriptive models of supervisory control

Human performance has been described by a wide variety of normative models, including decision theoretic, information theoretic, and control theoretic models [8]. In particular, human capabilities for performing manual control are fairly well understood, and a variety of normative models which quantitatively predict performance have been constructed for a variety of tasks. Such models use classical or modern control techniques.

Supervisory control has proven extremely difficult to model, because it includes a combination of control, decision making, and information processing tasks, as well as elements which do not fit neatly into any of these categories. At present, we must be satisfied with qualitative descriptive models. Although such models do not give quantitative descriptions of performance, they are useful in the design of supervisory control systems. Descriptive models are also useful in formulating experiments involving supervisory control systems.

Sheridan [6] gives a four dimensional qualitative descriptive model of supervisory control. These four dimensions will be described below.

1.4.2 Dimension 1: the function of the human operator

People perform a variety of functions in a supervisory control system. Sheridan [6] describes five distinct functions for the human operator, as listed in figure 1.1.

Planning and teaching must occur for all computer-based systems. In an autonomous system, these two functions are performed off-line, long before the machine performs the task. For a supervisory controlled telemanipulator, the planning and teaching should be extremely interactive. The operator must be able to continue planning and teaching even as the machine operates in the remote environment.

A supervisory control system also contains facilities that allow the operator to monitor the remote system and intervene if necessary. The operator also learns about the remote environment and how the teleoperator functions. This allows the operator to improve how well he performs the first four functions. These last three functions, monitoring, intervention, and learning, are not seen in automatons.

Generally, these five functions are performed iteratively, so adequate two-way communication between the human operator and the system is essential.

1.4.3 Dimension 2: the knowledge-rule-skill heirarchy

Rasmussen [9] has proposed a heirarchical model for human behavior in the control of complex systems. In general, the elements of this

1. PLAN

    - Be aware of tasks, evaluate resources
    - Decide on goals and tradeoffs among goals
    - Decide on strategy, including logic of authority between man and machine
    - Decide what is to be considered abnormal behavior, including automatic recovery from trouble

2. TEACH

    - Estimate what computer system knows of the situation
    - Decide how to instruct computer system
    - Test instructions in simulation, either with a mental or computer model
    - Initiate execution of instructions

3. MONITOR

    - Decide on what to observe and set up monitoring aids appropriately
    - Observe displays, looking for abnormal behavior
    - Make minor adjustments as automatic control continues
    - Diagnose abnormalities

4. INTERVENE

    - Decide when continuation of automatic control should stop
    - Implement appropriate recovery procedures
    - Initiate new planning, teaching, or monitoring as appropriate

5. LEARN

    - Collect salient data over repeated runs and draw inferences about how the system may be improved
    - Allow for serendipitous learning
    - Periodically take stock of learning and modify system appropriately
    - Develop understanding and trust of the system

Figure 1.1 Functions of the human operator as supervisor. Modified from Sheridan.

heirarchy may be found in each of the five functions listed in the previous section, so this heirarchy may be considered to be an independent dimension of the descriptive model.

Rasmussen's model decomposes tasks performed by people into three components. These will be listed from least abstract to most abstract.

1. Skill-based behavior: the operator continuously responds to changes in the system state. Manual control activities are examples of skill based behavior.

2. Rule-based behavior: the operator can recognize patterns or combinations of system states or the environment through previous experience or training.

3. Knowledge-based behavior: for each of the five functions of the human operator, the person must understand problems, set goals, and understand what to do next.

1.4.4 Dimension 3: Sensing, cognition, responding

Each combination of the first two dimensions may also be divided into three other components. These elements may be continuous in the case of skill-based behavior, or discontinuous in the case of rule or knowledge-based behavior.

1. Sensing: The human operator receives information from the computer system through displays, through a video system, and from other people or from the environment.

2. Cognition: The operator makes decisions based on sensed information. Current man-machine systems theory holds that operators build internal models of the processes they are controlling. These internal models are built through experience and training, are updated through the information sensed by the operator, and are used to formulate the proper response.

3. Responding: The operator can manipulate controls such as switches, joysticks, etc., or he may invoke or create programs.

1.4.5 Dimension 4: Different tasks

In the cases of process control, flight systems, and power plant control, the human operator must supervise many subsystems simultaneously. In the case of underwater telemanipulation, the different tasks include supervisory control of the shipboard equipment, the vehicle, and the manipulation task.

1.5 Design Goals in Terms of the Descriptive Model

Supervisory control may be seen as a process whereby a human operator offloads some of the skill and rule based behavior to the computer system [9],[6]. This concept will be used to generate some design goals for the system. For each of the five functions of the human operator, planning, teaching, monitoring, intervening, and learning, methods for off-loading elements of the task to the computer system will be discussed.

1.5.1 Planning

Planning is primarily knowledged-based, usually involving those parts of a task which cannot be broken down into rule and skill-based behavior. Planning goes on mostly in the operator's head. In planning, the human operator makes use of his knowledge of the task and his knowledge of the capabilities of the system. In planning, the operator decides which resources should be used and on the overall goals of the system.

In terms of the design of a supervisory telemanipulator, the operator's planning activity may be supported by making the system's capabilities and limitations well known to the operator. Later, graphical display assist will be presented to aid in this process.

During the planning phase, the operator may also describe rule-based behavior concerning when the system should decide it has failed and turn to the human operator for help. The design should include these facilities.

1.5.2 Teaching

Teaching involves knowledge, rule, and skill-based behavior on the part of the human operator. During teaching, the operator describes skill and rule-based elements of the manipulation task to the computer system.

From the point of view of system design, the human operator should be provided with three types of tools for teaching:

1. The operator must have some mechanism for describing movement in order for the system to perform the skill-based behavior. In this thesis, a system for defining movements in a variety of ways will be presented.

2. The operator must have some mechanism for combining movements together with logical or arithmetic operators. This facility allows the operator to describe rule-based behavior. Sensing routines, structured programming concepts, and extensibility will be used for this purpose.

3. The operator should have some mechanism for trying out a procedure after one has been designed, preferably without committing the remote hardware. A graphical display aid will fill this role.

1.5.3 Monitoring

Monitoring is primarily knowledge and rule-based on the part of the human operator. During the planning and teaching stages, the human operator has already off-loaded the rule-based and skill-based monitoring behavior that he can describe and for which the system has the appropriate sensors. The human operator must then monitor the system for the types of problems that the system itself cannot diagnose.

The computer system can aid the human operator in monitoring. Real-time graphical representations of the system state will be presented. These graphical aids may be controlled by the human operator to yield the types of information desired about the system state. Symbolic monitoring aids, where the operator requests information in alphanumeric forms will also be included.

1.5.4 Intervention

After the operator has detected and identified a problem with the help of the monitoring aids, he may also use the computer system to intervene and get the manipulator out of trouble. Deciding when to intervene and how to intervene is often a knowledge-based activity on the part of the human operator, but may be rule-based for anticipated types of trouble.

During intervention, the operator may intervene directly or he may decide to assign some of the recovery activity to the system. The options presented in this thesis include switching the system into manual control, or invoking several computerized recovery routines.

1.5.5 Learning

Learning can be knowledge-based, rule-based, or skill-based on the part of the human operator. The system can aid in this process in similar ways to how it could aid in planning: by always showing the human operator in clear terms how the system responds to instructions and the environment under varying conditions.

1.6 Experimental goals

Performing experiments concerning supervisory control is difficult. The diversity of human activity in supervisory control makes the process difficult to model. Similarly, it is difficult to ask questions that are specific enough to make good experiments yet still have some meaning outside the exact conditions tested. It is impossible to experimentally evaluate each design decision in any practical, useful system.

Many types of experiments are performed to study man-machine systems. While the question of designing experiments will be discussed in detail in Chapter 5, a summary is presented here.

At one extreme are controlled experiments from the tradition of experimental psychology. All independent psychological factors are first identified. Then, experimental trials are performed for each combination of these independent factors. With proper design, conclusions about the importance of the independent factors and their interactions may be made on a quantitative basis. Varying all independent factors takes a large effort for even relatively simple situations, but the potential benefit of understanding how the independent factors interact (or don't interact) allows the results to be extrapolated to other situations not tested. Several investigators have pursued this course concerning telemanipulation [10],[11].

At the other extreme are purely descriptive experiments. These have commonly been performed using real underwater telemanipulators in actual underwater tests [4], [5]. In these experiments, no attempt is made to isolate and vary all independent psychological factors. Generally, a series of tasks is chosen. The results state whether the task could be completed, how long it took, and some measure of performance or efficacy (often subjective).

In this thesis, controlled experiments will be pursued. The experiments will be of two different types.

The first type of experiment will concern human performance in those skill-based activities that cannot be off-loaded to the computer system and are deemed critical in making the supervisory control work. The methods of experimental psychology are very effective in investigating these issues and will be applied as rigorously as possible in these experiments. Quantitative results will be obtained.

The second type of experiment will concern overall system performance for different supervisory configurations. Although "supervisory configurations" is multidimensional in a psychological sense, it will be treated as a single independent variable. A task has been designed that reflects elements useful for inspection of complex welds. Performance in this task can be precisely defined in terms of two error measures. The performance over changing supervisory configurations will be analyzed quantitatively.

Chapter 2

MOTION CONTROL OF A TELEMANIPULATOR

2.1 Introduction

The previous chapter discussed what a supervisory control system is, the reasons supervisory control is needed, and the relationship between supervisory control systems and other types of autonomous and semiautonomous systems. This chapter discusses the elements that must be considered to control the motion of an underwater telemanipulator. As each of these elements is presented, specific solutions will be described for the two manipulator systems described in the previous chapter.

For the two systems considered here, control of the movement of the arm will be based on high level position and velocity specification. Other types of systems have been built where an arm can be controlled by specifying the desired force between the end effector and the environment [13], or possibly force on some degrees of freedom, and velocity on others [14], [15]. However, force control was not considered in this thesis for two reasons. First, the required instrumentation is not currently available. Recently, wrist force-torque sensors have begun to appear on the market [16], but application of this technology to the underwater environment is still a step away. Second, most of the tasks currently being investigated for remote controlled underwater manipulators do not require force control

[4]. These tasks include both inspection and simple maintainence tasks. A review of these tasks can be found in Appendix G.

Control of the mechanical impedance of a manipulator arm may prove to be more effective than force control for controlling the interaction between a manipulator and the environment, with less sensing required [17]. Experimentation with such a system is now underway [18].

2.2 Manipulator Kinematics

The choice of kinematics for a manipulator bears directly on the size and shape of the workspace, the number of ways a given position can be reached, and the suitability of the arm for computer control [19]. Therefore, manipulator kinematics will have a large impact on the design of a supervisory manipulator system and on what level a human operator can communicate with the system.

2.2.1 Number of degrees of freedom

A basic design decision concerns the number of degrees of freedom for the arm. Six independent degrees of freedom are necessary, although not sufficient, for the manipulator hand to be positioned generally within the manipulator's workspace. This allows the human operator to communicate with the system on the "ideal effector" [20] level. This means that descriptions of the hand motion can be given without requiring the operator to consider the joint movements required to accomplish the movement.

As the number of degrees of freedom is decreased from six, communication at the ideal effector level becomes more difficult. Translations of the manipulator hand will require a change in the hand orientation. A simple translational command like "move 10 cm to the right" may require that the orientation of the hand change as well, depending on the kinematics of the arm and the current configuration of the arm. In general, it will be more difficult to make the arm's kinematics transparent to the human operator.

2.2.2 Relationship between degrees of freedom

It is not sufficient to have 6 degrees of freedom to have a manipulator that can be controlled on the ideal effector level. The choice of how those degrees of freedom are arranged is also of great importance. Reviews of this issue are given by Pieper [21] and Roth [22]. To summarize these results, the kinematic configuration chosen for a manipulator arm greatly influences the shape of the workspace, the number of hand orientations with which the arm can approach a given position, and the complexity of the computations required to solve the kinematics problems.

Manipulator kinematics are usually classified using notation developed by Denavit [23] and developed for manipulators by Roth and Pieper [21]. This notation is described in Appendix H.

2.2.3 Influence of kinematics on high level velocity specifications

Whitney, [7] designed a method for specifying the cartesian velocities of the end effector of a manipulator called Resolved Motion Rate Control. The method has been implemented as a manual control technique, with the human operator specifying the cartesian velocities with an analogic controller [24].

The method involves inverting the jacobian or shape matrix. The jacobian relates the joint velocities to the cartesian hand velocities. A problem encountered with this method is that at some configurations, the jacobian becomes ill-conditioned and therefore difficult to invert. This is not a failing of the computational method, but a reflection of the fact that the arm has kinematics that make cartesian velocity control very difficult to achieve at that configuration.

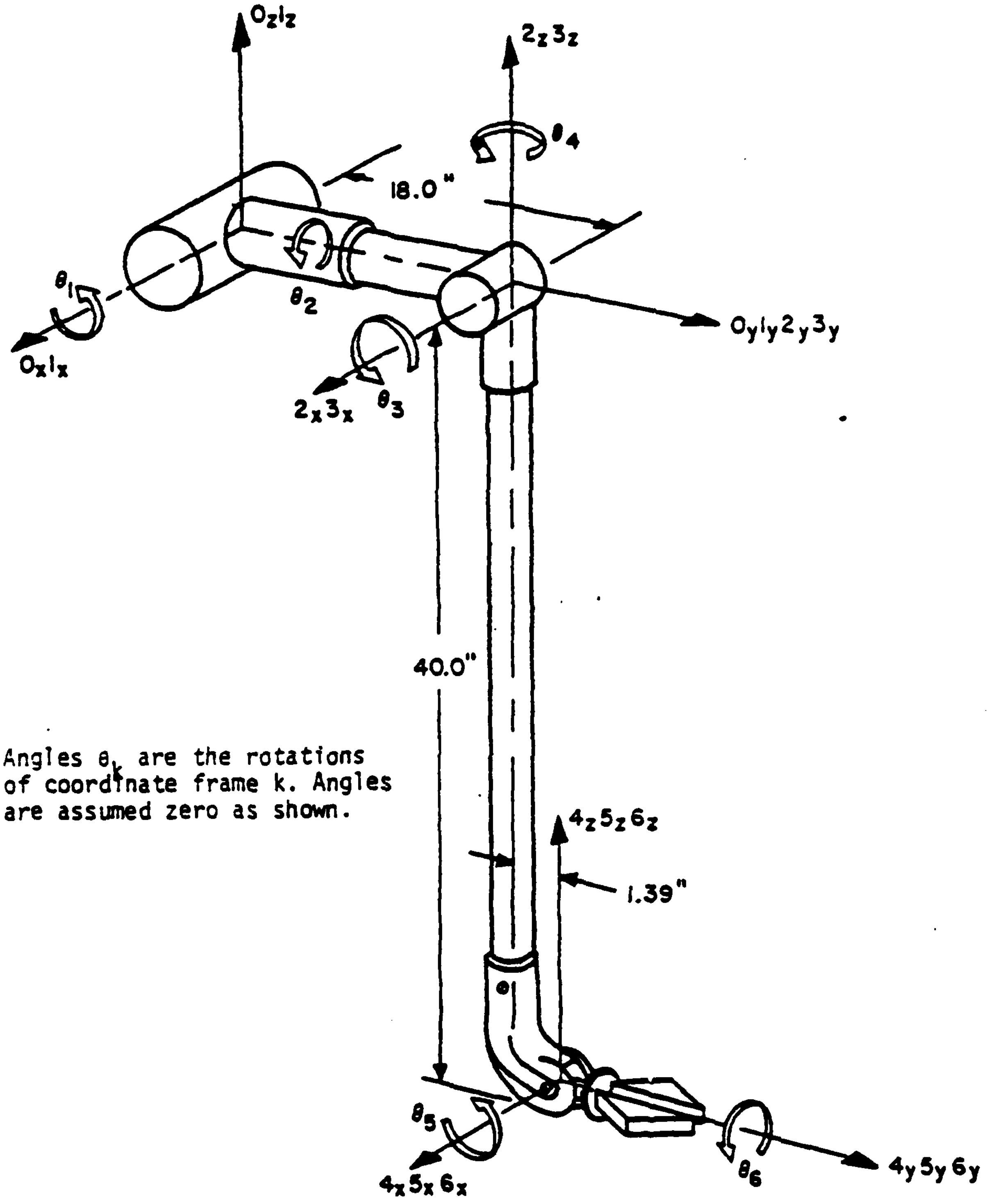
2.2.4 Influence of kinematics on high level position specifications

If a manipulator arm is to be controlled from high-level position specifications, two kinematic problems are usually solved. The first of these,the forward kinematics, refers to the problem of determining the position and orientation of the manipulator hand given the joint angles. The second problem is called the inverse kinematics problem and refers to obtaining the joint angles that will bring the manipulator hand to a given position and orientation.

The forward kinematics are always solveable in a direct, closed form fashion. Usually this problem is solved using homogeneous transformations [25]. Recently, quaternions have been shown to be more economica` from a computational point of view [26]. Quaternions will be discussed in chapter 3 with other methods for representing rotation.

A solution to the inverse kinematics problem is not unique and no general solution has been designed [21]. In particular, the relationship between the layout of the arm degrees of freedom and the complexity of the inverse kinematics solution is a major design consideration for any computer controlled manipulator. For example, six degree of freedom manipulators are generally designed so that the last three axes of rotation intersect at a point. This design feature allows the six dimensional problem to be broken down into two three dimensional problems, which are much easier to solve than the general 6 degree of freedom problem. Roth has summarized the solveability of manipulators by the relationship between the degrees of freedom.

Kinematic design criteria for manually controlled and computer controlled manipulators can be different. For example, the Argonne E2 master-slave manipulator has a separation of about 3 centimeters between the fourth and fifth degrees of freedom (figure 2.1). This separation is included to prevent gimbal-lock when a person is moving the master and results in improved performance under manual control. However, this small separation makes solution to the inverse kinematics more difficult. An approximate solution method has been designed and



Angles θ_k are the rotations of coordinate frame k. Angles are assumed zero as shown.

Figure 2.1 Argonne E-2 Manipulator (from Brooks)

implemented (Appendix B). However, this solution requires almost twice as much computation compared to the solution for the arm without the offending 3 centimeter separation. The method also results in an approximate solution, although more exact results can be obtained at the expense of more computation.

2.2.6 Kinematics of the NOSC manipulator

An underwater manipulator was built at the Naval Ocean Systems Center, San Diego, as part of a program to develop free-swimming remotely controlled vehicles. Personnel at the MIT Man-Machine Systems Laboratory have been collaborating with NOSC personnel to design a control sytem and man-machine interface for the manipulator.

The NOSC manipulator has 5 rotary degrees of freedom, plus grasp. This is sufficient for grasping most objects, positioning sensors, cleaning welds using a water jet, or manipulating a camera for close-up visual inspection [27]. Coordinate frames corresponding to each degree of freedom are shown in figure 2.2. Using the standard kinematic notation described in Appendix H [19], this manipulator may be classified as:

a2s4

Five degrees of freedom allow the hand to be positioned generally over a reasonable workspace. Only two orientations may be specified, the value of the third orientation will be determined by the kinematics. In general, two combinations of the first three joint angles will place the

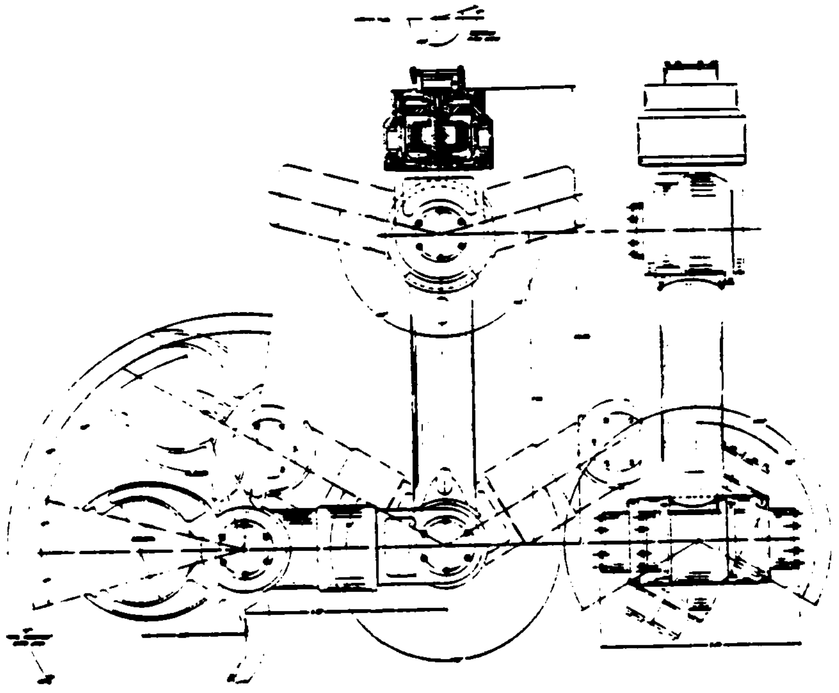


Figure 2.2 NOSC manipulator (from Bosse)

wrist of the manipulator at the same position. These two solutions will be called "over" and "under". The possible orientations the wrist may assume depends on which of these two solutions has been chosen, as shown in figure 2.3.

The positions achievable by this arm are also limited by the allowable range of motion of each rotational joint. The range of possible positions for the wrist point is summarized in figure 2.4 for both "over" and "under" joint space solutions.

The range of motion of the wrist joints also limits the orientations achievable from a given wrist position. The hand may point in any direction within a cone. The size of the cone is defined by the maximum rotation of the last joint. The direction of the axis of symmetry of the cone is determined by the first three joints. This is summarized in figure 2.5, and the orientation of this cone is shown for various positions in figure 2.6.

The inverse kinematics solution for the NOSC arm is described in detail in Appendix A. For a given desired cartesian hand frame, the algorithm generates the five joint angles for the desired solution for the first three joints (either "over" or "under"). The rotation about the hand axis specified by the desired hand transform is ignored, reducing the six degree of freedom quantity to five degrees of freedom.

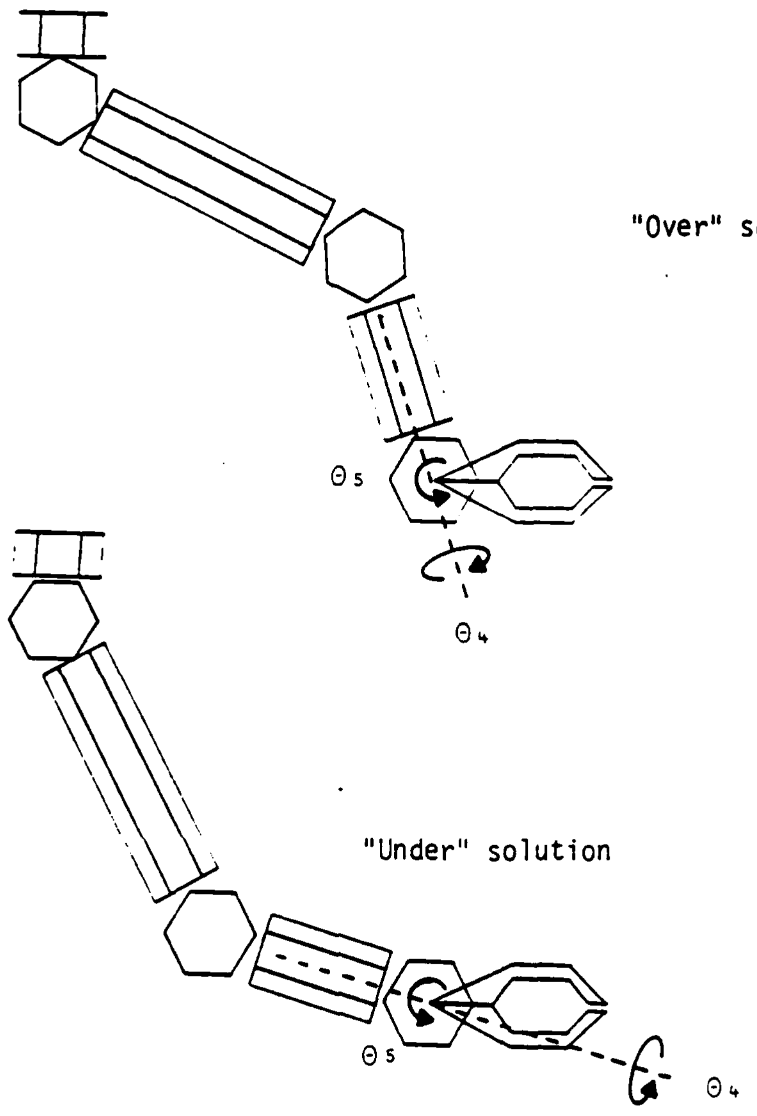


Figure 2.3 The influence of position on orientations for the NOSC arm. Two combinations of the first three joints can position the wrist at the same location. Different wrist orientations are possible from these two configurations.

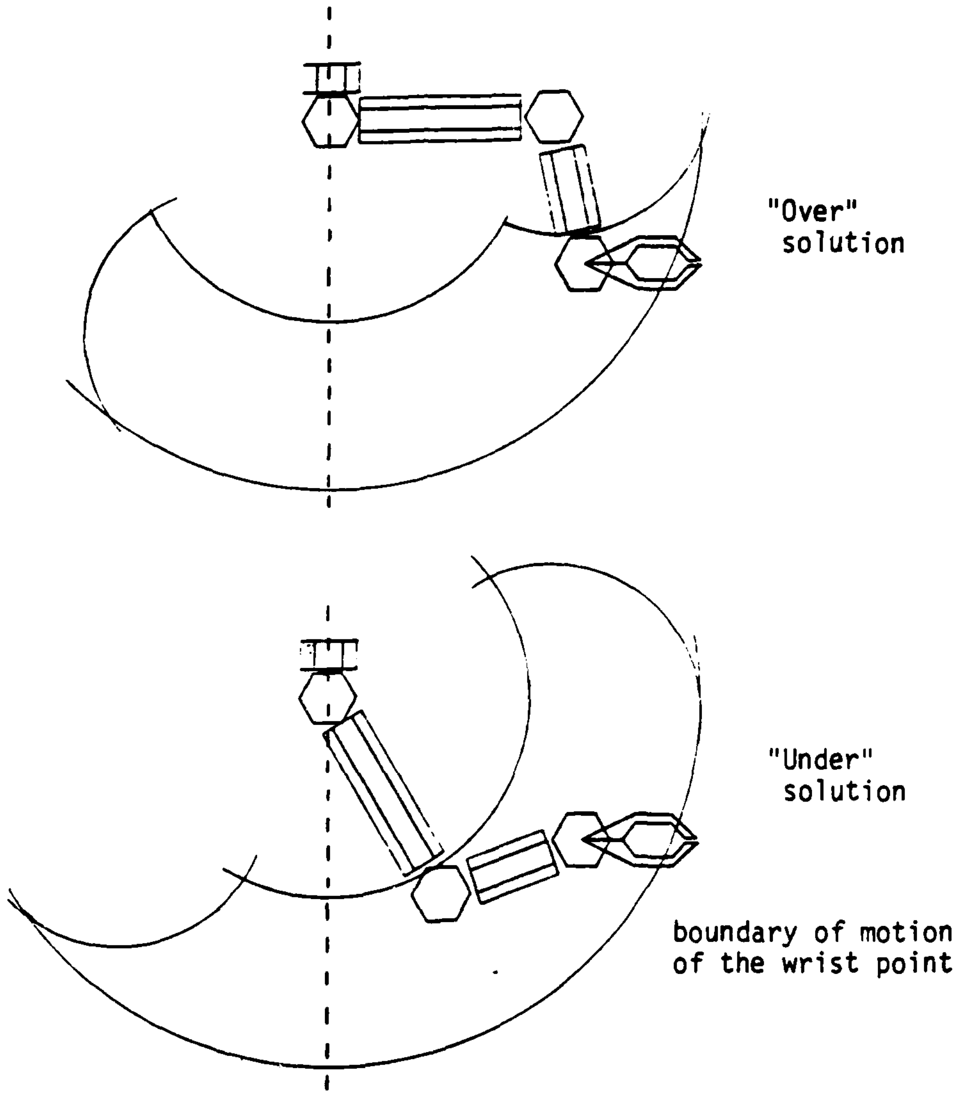


Figure 2.4 The working volume of the NOSC manipulator. The size and shape depend on the choice of joint space solution. The volume is rotationally symmetric about the dashed line.

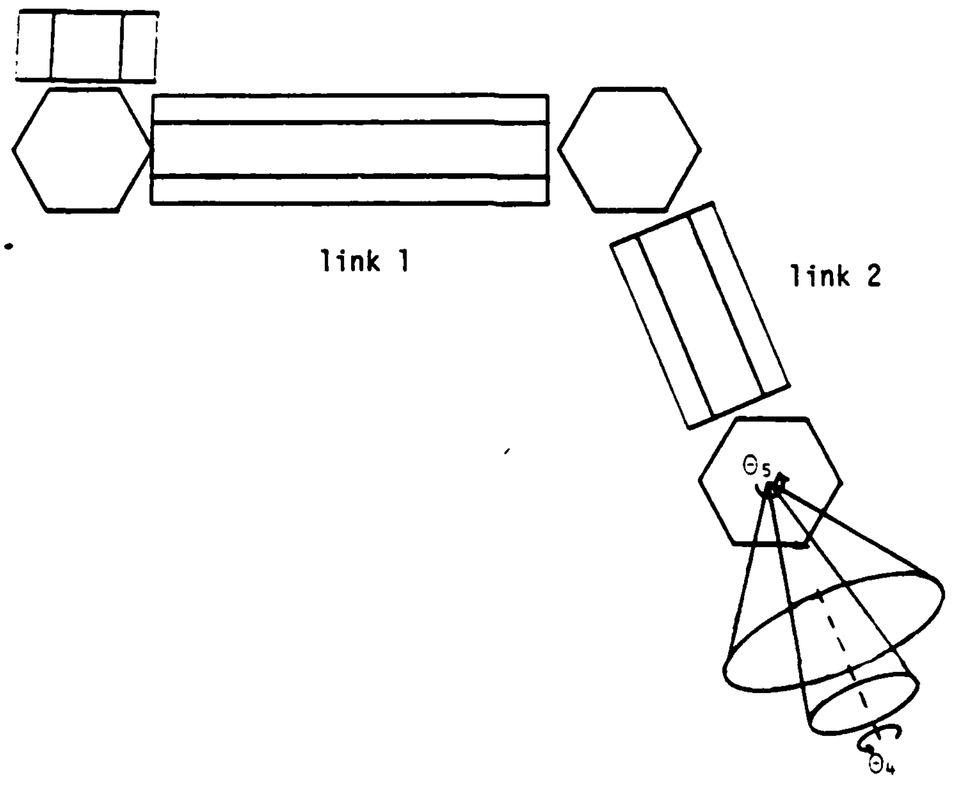


Figure 2.5 Possible wrist orientations for the NOSC arm. For a fixed value of Θ_5 , rotation about Θ_4 causes the hand to sweep out a cone. The maximum half angle of the cone is determined by the maximum rotation of Θ_5 (75°). The axis of symmetry of the cone lies along the longitudinal axis of link 2.

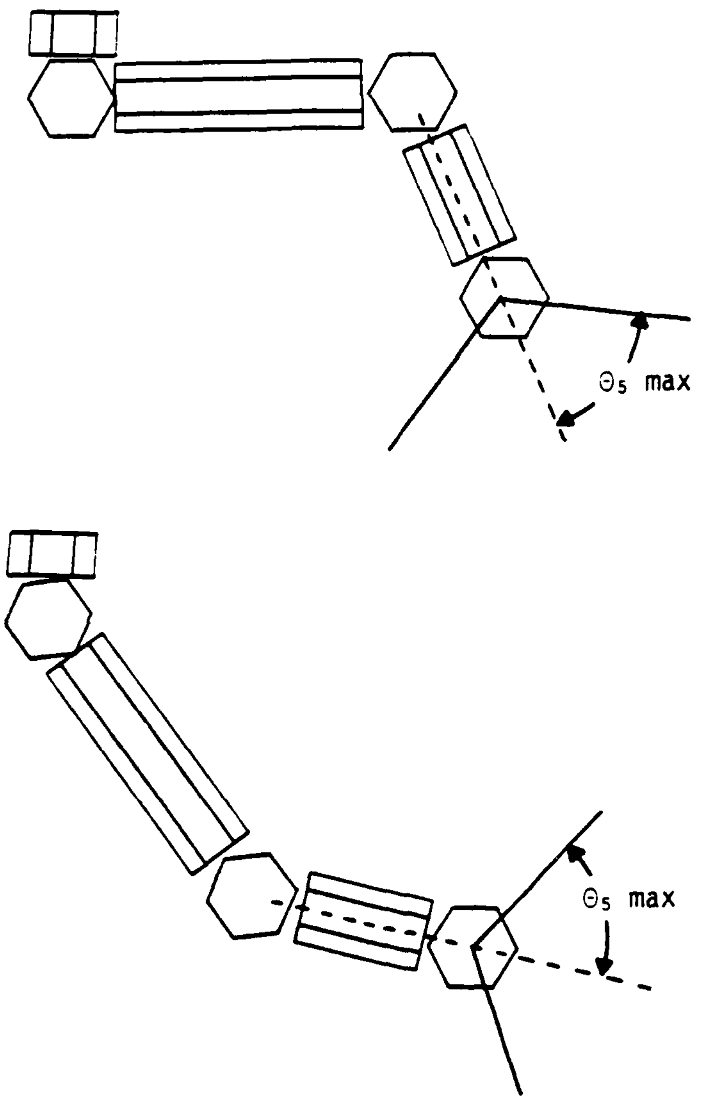


Figure 2.6 Restriction on hand orientation for the NOSC arm. The range of hand orientations depends on both the position and choice of joint space solutions.

2.2.7 Argonne E-2 manipulator kinematics

An Argonne E-2 manipulator is used for supervisory control experiments at the MIT Man-Machine Systems Laboratory. The manipulator was modified by Brooks [10] to be controlled in a hybrid mode, with setpoints to the analog position servoes specified by a computer.

The E-2 manipulator has six rotational degrees of freedom, plus grasp. This arm was designed to operate in a force reflecting master-slave mode for remote handling of dangerous materials. The arm is shown in figure 2.7, with coordinate frames identified for each degree of freedom. Using standard kinematic notation, this arm is classified as:

s2s4a4

Six degrees of freedom allow the arm to be positioned and oriented generally, although there are limits to the rotation of each joint. Because of the limited rotation of the first three joints, only one valid joint space solution exists for a given wrist position. The range of reachable wrist positions is summarized in figure 2.7.

The inverse kinematics solution for the E-2 arm is described in detail in Appendix C. As described earlier, the solution is not straightforward since the last three axes of rotation do not intersect at a point. The algorithm solves the inverse problem for a similar arm to arrive at a first approximation, then transforms the approximate solution to the solution for the E-2.

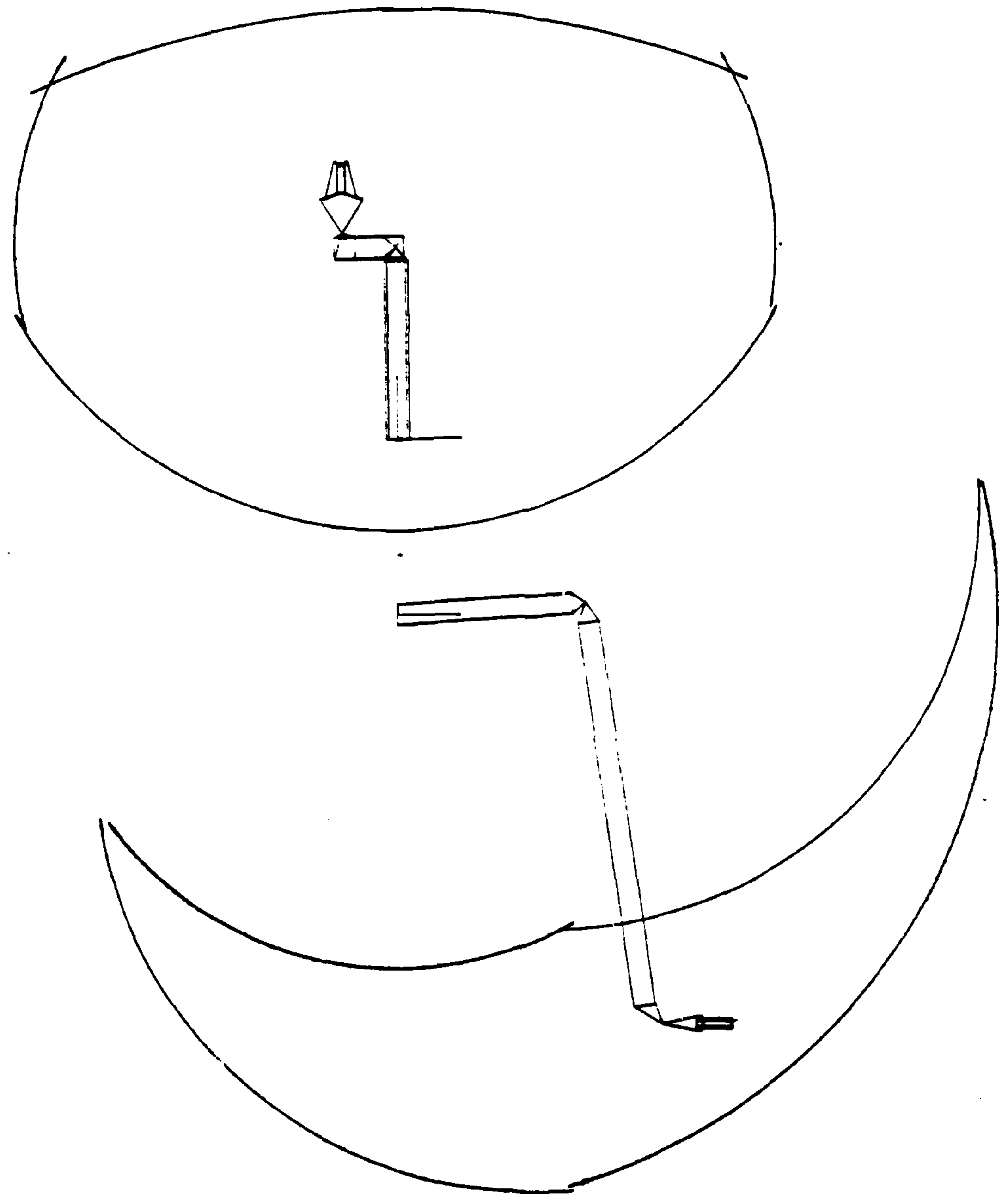


Figure 2.7 Working volume of the Argonne E-2 manipulator

2.3 Manipulator Dynamics

2.3.1 Importance of the dynamics problem

After a desired joint space configuration has been computed from the manipulator kinematics, the problem of getting the manipulator arm to that configuration remains. Since manipulators are nonlinear systems with a high degree of coupling between joints, an analytical solution to this problem is formidable. This section will review various methods and comment on their suitability to the context of this thesis.

First, the desired joint movements can be computed from some high level specification, the measured values of position and velocity and knowledge of the kinematics. Then, the actuator torque commands must be computed.

These torques must be computed to bring the dynamics of the arm under control. The computation may explicitly model the dynamics of the arm, or simple approximations may be used, in which case the system will rely heavily on feedback.

Classical Lagrangian methods can be used to solve this problem. Such a solution would model inertial, centripedal, coriolis, and gravitational forces. However, the first implementation of this solution for a six degree of freedom manipulator required a prohibitive amount of computation each time step [25]. This result led to the search for less computationally demanding methods. A summary of these results is given by Hollerbach [28].

Instrumentation has a large influence on the performance of a control system. Position measurement is an absolute necessity if the computer is to aid the human in any substantial way. A velocity measurement will greatly improve position response time and allow velocity to be regulated as well. Velocity estimates can be used in lieu of measurement, depending on how well the system can be modelled. Velocity estimates may be obtained through differentiation (with the accompanying noise problems), or through an observer, from modern control theory.

2.3.2 Approximation methods

Nearly all manipulator control systems in use today use approximation methods. These methods generally model each joint as a fixed inertia with some damping. Coriolis and centrifugal forces are ignored, and feedback is used to compensate for these unmodelled forces [29], [30]. These methods are also employed in most industrial robots today, where each joint is controlled by a high bandwidth, independent position servo. Experience has shown these methods to be sufficient, although the system will perform well below its full capability [31].

2.3.3 Recursive Dynamic Methods

Recently, the efficiency of the original Uicker-Kahn Lagrangian formulation has been improved by several orders of magnitude. By formulating the dynamics solutions recursively, the solutions can run at practical speeds on relatively modest (i.e. PDP 11) computers.

Luh, Walker, and Paul [32] have implemented such a method. The method uses a recursive Newton-Euler dynamics solution to form an inverse plant model. Recently, Silver [33] has shown that Lagrangian methods can produce an equivalent result.

2.3.4 Model Reference adaptive control

Another approach to this problem is to use adaptive control methods. Several investigators have been using model reference adaptive control methods to control manipulators [34], [35]. These methods have the advantage that only simple, general models of the arm are used. The method is capable of adapting when the load on the manipulator changes. This method could be useful for a high performance underwater manipulator, when the drag forces could influence performance and be difficult to model.

2.3.5 Controlling the dynamics of the NOSC arm

Each rotational joint of the NOSC arm contains a DC torque motor, a harmonic drive gear train, and a potentiometer enclosed in a sealed unit which will be filled with oil when the unit is put in the water [27]. The motors are driven by voltage controlled linear amplifiers.

The control method used considers the joints to be independent. Position servoes regulate the position of each joint. Moments caused by interaction of the links are dealt with implicitly through local feedback. This approach was chosen for several reasons:

1. computational resources were limited;

2. velocity measurements were not available;

3. the large reduction ratios of the harmonic drives cause the inertia of the motors and gear drive to be the dominant inertias in the system (see Appendix C);

4. the requirement that the arm move slowly to minimize disturbances on the lightweight vehicle carrying the arm.

The last two reasons mean that the dynamics of the system are dominated by fixed inertias, friction, and gravity.

The servo loops were implemented in digital form. Except for the two joints at the shoulder, each loop is a proportional position servo, with a maximum value allowed for the computed voltage. At the servo level, the joint positions are always checked to insure that the joints are not driven out of their range.

The joints at the shoulder have additional features. The inertia of the first joint changes as the arm extends, so scheduled gains based on configuration were used. The second joint is influenced by gravity depending on the values of the second and third joint, so integral control was added. These controllers are described in more detail in Appendix C.

The first joint, although influenced by gravity has zero steady state position error for torque disturbances because of the action of the integral controller [36]. The remaining joints have very low steady state position errors for torque disturbances because of the high gear reductions.

2.3.6 Control of Dynamics of the E-2 manipulator

The dynamics of the E-2 arm are controlled by the original analog servo controllers supplied with the arm, and modified by Brooks [10]. When under computer control, the servoes are type 0 position servoes with both position and velocity feedback. These servoes do not compensate for gravity, but all links are counterbalanced mechanically.

2.4. Generating Trajectories From Cartesian Specifications

This section describes various methods that have been used to generate arm trajectories from cartesian space specifications.

2.4.1 Specifying the hand cartesian velocity trajectories

The most common method used to manually control hand cartesian velocities is Resolved Motion Rate Control [7]. The influence of arm kinematics on this method was discussed earlier. The method was originally designed for manual control of teleoperators and prostheses.

General cartesian velocity control is not possible near kinematic singularities. The problem lies not with the computational methods, but with the kinematics of the arm. Any system which attempts to continuously control carte-ian velocity will have this problem.

2.4.2 Specifying hand cartesian position and velocity trajectories

2.4.2.1 Intermediate cartesian frames

Most systems currently in use employ intermediate cartesian frames to produce straight line motions to a goal specified in cartesian space. These methods are summarized in figure 2.8. A number of intermediate cartesian frames lying along the straight line path are computed. The inverse kinematics are solved for each of these cartesian frames producing intermediate joint frames. Interpolation is then done in joint space between joint frames.

This method may also be used for cartesian velocity control. The ratio of the di͵tance between frames to the time between frames will approximate the translational velocity of the manipulator hand. Near kinematic singularities, the trajectory may depart more from a straight line, but a solution will always be found.

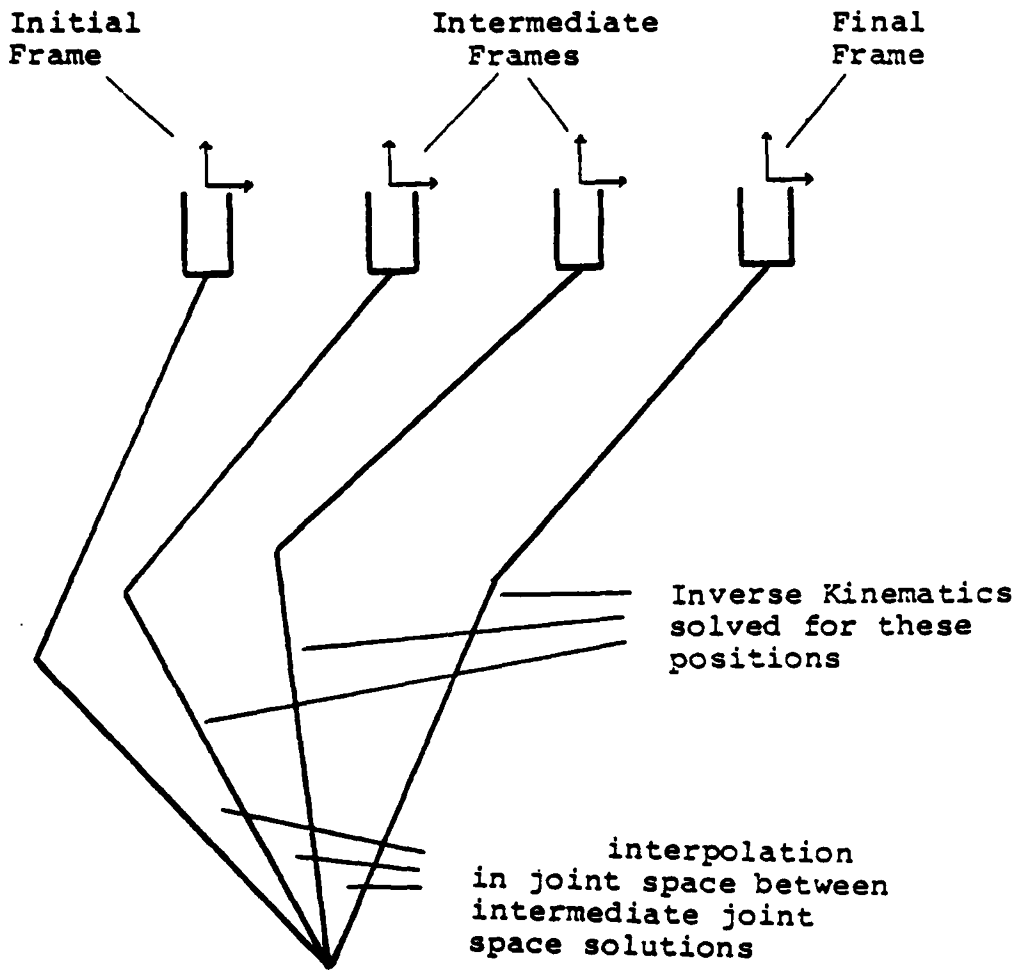

Initial Frame
Intermediate Frames
Final Frame
Inverse Kinematics solved for these positions
interpolation in joint space between intermediate joint space solutions

Figure 2.8 Intermediate cartesian frames. Cartesian trajectories may be planned by computing a number of intermediate frames, and performing joint space interpolation in between.

There are several variations on this method. The major differences are:

1) how the number and position of intermediate frames are chosen,

2) what "straight line motion" means when the hand orientation changes during the motion. While the tip of the hand or tool should follow a straight line, a linear interpolation of the hand orientation angles will depend on how those angles are defined.

Paul [37] implemented such a system that included straight line motion, smooth transitions between straight line sections, and smooth transitions to moving coordinate systems. This system interpolated at constant frequency (10 hz).

Paul defined "straight line" motion involving changes in hand orientation as uniform rotation about 2 axes. Smooth movement between straight line segments was accomplished using quadratic interpolation . The method provides for a transition which begins and ends with zero acceleration, resulting in no discontinuity in velocity.

Near singularities, joint space velocities required to produce a moderate cartesian velocity can grow very large. Paul solved this problem by checking if any joint velocities exceeded prespecified limits. If such a condition existed, all velocities were scaled down so that all joints could keep up.

These techniques have been further refined by Taylor [26]. Taylor's methods use a different interpretation of straight line motion for hand orientation which reduces the amount of computation. Taylor specifies hand orientation by means of a quaternion. This representation considers a change in orientation as a single rotation about an axis:

$$R = Rot(n,\theta)$$

where n defines the direction of the angular displacment

θ defines the magnitude

This representation makes the determination of intermediate frames simpler, since n remains fixed and only θ is varied. For complex rotations, this may be more difficult to understand than Paul's method.

Taylor's method of computing the intermediate frames allows the deviation from the exact straight line path to be specified. Intermediate cartesian frames are computed recursively at intervals which keep the deviation from a straight line below these limits, rather than at constant intervals in time or cartesian space. The method requires that trajectories be precomputed, but saves computation. This scheme of choosing intermediate frames is called Bounded Deviation Joint Paths.

2.4.2.2 Resolved acceleration control

Cartesian position and velocities can be specified directly on a continuous basis in a method implemented by Luh et. al. [38]. The desired joint accelerations are continuously computed based on the errors between commanded and measured cartesian positions and velocities. These joint accelerations are then used to compute the necessary joint torques using the recursive Newton-Euler dynamics formulation discussed earlier.

The method allows cartesian accelerations, velocities, or positions to be specified, either separately or in combination. No pre-planning is required and the computational requirements are compatible with fairly modest computers. The same control system is equally useful for both resolved motion manual control and computer control modes.

Since this method attempts to control cartesian velocity continuously, problems similar to those in Resolved Motion Rate Control arise near degenerate arm configurations.

2.4.3 Cartesian Path Control for NOSC and E2 Systems

Both systems can execute movements to positions and orientations defined in a cartesian sense. These cartesian descriptions may be defined relative to the manipulator base, the manipulator hand, or any other reference frame. The methods to describe these motions will be covered in Chapter 4. This section will describe how the trajectories are computed from a general cartesian description.

In order that the computational load be reduced to a minimum, motions are planned with an endpoint and a minimal number of intermediate cartesian frames. These intermediate frames are specified by the operator through a variety of means presented in Chapter 4. In between the cartesian frames, joint space interpolation is performed. Straight line motions involving translation only may be programmed by specifying a sufficient number of intermediate frames.

In addition to being economical from a computational point of view, this method has the advantage that fewer problems from kinematic singularities are encountered. If an ill-conditioned region lies between two intermediate frames, no problems are created since no kinematic computations are performed between intermediate frames. Control of cartesian velocity is less precise, however, especially near singularities.

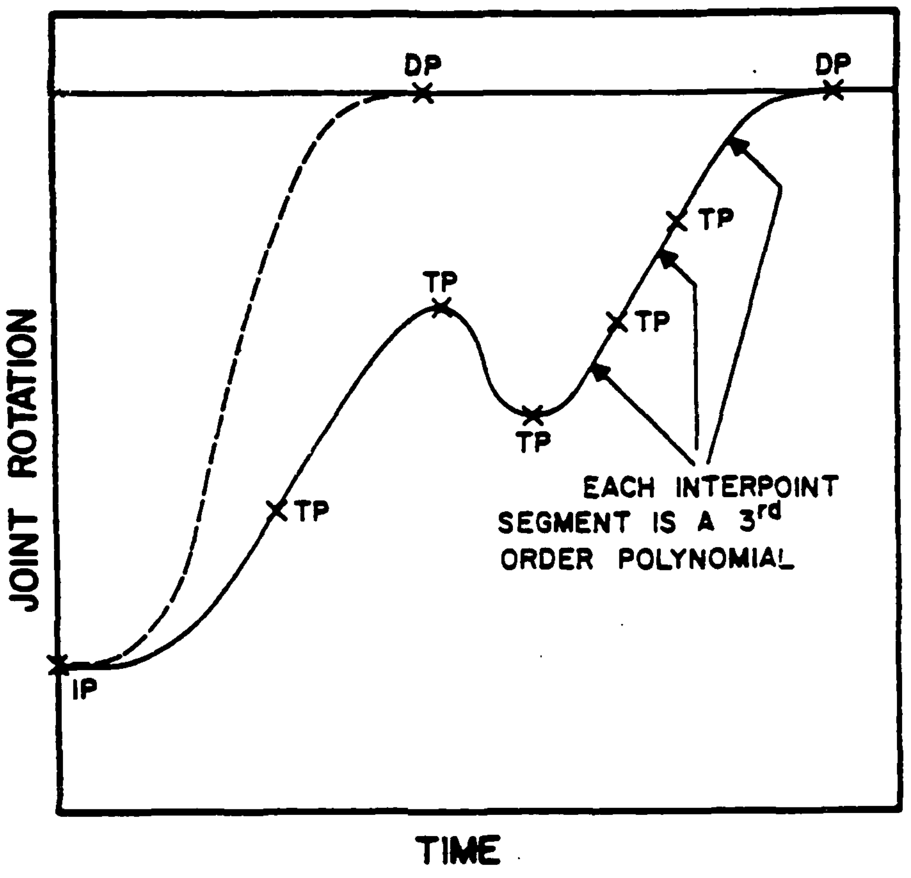
The joint space solutions for the intermediate frames may be precomputed and placed in a first-in-first-out (FIFO) queue. The trajectory generation system will then remove the next joint vector from the queue as the previous joint space vector has been reached, and generate a joint space trajectory to the next joint space vector. The trajectories may be specified as continuous, or the system may be made to stop at each intermediate frame.

For the NOSC manipulator, a first order (straight line) joint space interpolation is used. The discontinuity of velocity between segments is not apparent because of the low bandwidth of the servoes.

The E2 manipulator has much higher bandwidth than the NOSC arm, primarily due to the presence of velocity feedback and the gravity compensation through counterbalancing. A smoother joint space interpolation is required, or the servoes will generate very high motor currents at the velocity discontinuities. Third order polynomial interpolation, as implemented by Brooks [10] was used. An example is shown in figure 2.9.

2.5 Conclusion

This chapter has considered methods for controlling the velocity and position of a manipulator arm. In addition, the implementation of the two systems developed for this thesis were described.



TP-THROUGH POINT

DP-END POINT

IP-INITIAL MANIPULATOR POSITION

Figure 2.9 Joint space interpolation for the E-2. Third order polynomial joint space interpolation is used to generate trajectories between intermediate frames (modified from Brooks).

CHAPTER 3

OPERATOR INTERFACES FOR CONTROL OF MANIPULATORS

3.1 Introduction

The complexities of manipulator control described in Chapter 2 make the use of computers to aid a human operator desireable and sometimes necessary to get a particular job done.

A large number of systems exist for this purpose. Virtually all industrial robots and a few underwater telemanipulators have computerized functions to aid the operator in addition to simply controlling the dynamics. The simplest of these interfaces gives the operator the ability to teach fixed positions and paths. At an intermediate level, the operator is given a programming language with which to control the manipulator. At the most advanced level, the operator gives a high level description of a process, after which the system can plan all operations required to carry out that process. This final level is the subject of considerable research efforts in the artificial intelligence community.

The design of these interfaces is driven by many considerations. One of these is the nature of the task, including the environment in which this task must be performed. Another factor is the characteristics of the manipulator. A factor that is often neglected relates to the capabilities of the person controlling the system.

3.2 The symbolic-analogic mix

Controls for a manipulator system may be divided into two principal groups: analogic and symbolic [1].

Analogic controls are those in which some motion of the human operator is physically isomorphic to the response of the machine desired. Such controls include joysticks, master-slave controls, knobs and dials.

Symbolic control elements lack this isomorphism with the task. The simplest form of symbolic controls are the button boxes commonly used to control an industrial manipulator during teaching. More complex symbolic interfaces include programming language interfaces, which allow the operator to write descriptions of the task the manipulator should perform.

The distinction should be made between analogic controls and analogic teaching. Analogic teaching involves teaching by moving the manipulator. This movement could be effected with either symbolic or analogic controls. Analogic teaching has also been called guiding [39].

Analogic programming can make a system more easy to use. Such systems have the potential to allow skilled craftsmen rather than computer programmers to operate computer controlled machines [40]. If the operator knows what motions the manipulator should follow, these motions can be communicated to the system. This type of control has found great favor for programming industrial robots that must repeat fixed motions, as is common in materials handling, painting, and spot welding operations [41]. The Unimation Apprentice system allows an experienced welder to teach the robot to perform continuous seam welds through analogic methods [42].

The shortcoming of analogic teaching is that it is very difficult to program decisions. Integrating information from sensors to change how the arm moves is difficult in a purely analogic system. This problem has led to the introduction of symbolic elements into these systems. The symbolic interfaces range from simple teach boxes to programming language systems.

Brooks [10] has explored the analogic-symbolic mix experimentally in relation to telemanipulators. Brooks designed a supervisory control interface which combined analogic and symbolic control in a flexible manner. Brooks found improved performance in a variety of tasks over all forms of manual control except for master-slave with force feedback.

Grossman [39] has explored analogic programming or guiding to program assembly operations, and he gives a list of design considerations for combining analogic and symbolic teaching. Informal experiments with trained programmers showed that a combination of guiding and symbolic programming was clearly faster than a purely symbolic progamming system.

These previous studies indicate that a human operator should have both symbolic and analogic teaching tools for programming a telemanipulator. The nature of the symbolic portion of the interface will determine what type of operator can sucessfully teach the system. For example, a symbolic interface in which the operator expresses motions in terms of operations on matrices requires that the operator have advanced engineering training. Symbolic interfaces that allow the operator to express motions in more intuitive terms are also possible, although such an interface may be less general.

3.3 Control of the machine versus control of the process

A human operator can often make a machine perform a task by directly manipulating the state of the machine. In this instance, the human operator uses his insight into the task to insure that the prescribed machine states get the job done.

In process or task level control the human operator gives descriptions in terms of what the system should accomplish. The system must then infer the required machine level control to produce the desired change of state in the environment.

An alternative to the machine level-process level distinction classifies control on the basis of the sensors used in the control. Whitney [43] classifies tasks as exteroceptive or interoceptive, depending on the sensing required. This division is based on the physiological framework introduced by Sherrington and others [44]. This framework divides the body's sensors into three categories [45]:

Exteroceptive sensors are those which are involved in sensing the state of the external world or the interaction of the body with the external world. These sensors include vision, smell, hearing, and the cutaneous senses. These sensors are all at or very near the surface of the body.

Interoceptive sensors are those which sense deep internal body changes. These include the sensors in the gastrointestinal tract.

Proprioceptive sensors detect changes between the inner and outer body surfaces, in the subcutaneous tissues, the muscles and the tendons. The changes detected by these sensors can relate strictly to the internal state of the body, such as the function of the sensors that detect joint position during free space motions. But proprioceptive sensors can also relate to interactions between the body and the environment, much like the exteroceptive sensors. The possible function of the Golgi tendon organs in determining the weight of an object to be lifted [46] is an example of a sensor that detects the body's internal state but are used to gather information about the external environment. These sensors are sometimes grouped with the interoceptors.

Because of the dual nature of proprioceptive sensors, like force sensors, the process versus machine framework will be used here. This methodology categorizes the function in terms of the descriptions used for control rather than the sensors employed. The difference between machine level control and process level control may be summarized as follows: process level control allows the operator to specify what he wants done; in machine level control, the operator must specify how to do what he wants done.

In this thesis, machine level control will be achieved through descriptions of motion, in terms of position and velocity. These descriptions will be communicated by the operator through both symbolic and analogic means.

Process level control will be achieved by combining movement descriptions and sensing commands in a heirarchical structure.

3.4 Heirarchical Representations of Manipulation Tasks

A variety of methods have been used to represent manipulation tasks heirarchically. For machine based planning of manipulation tasks, such representations are used by a system to decompose high level goals into a sequence of low level procedures that will achieve the goal. Examples of heirarchical descriptions used for supervisory telemanipulation include Hardin's AND-TREE structure [47] and Sacerdoti's procedural nets [48].

Freedy, Shaket, and others at Perceptronics have modeled manipulation tasks using a system based on procedural nets. An example of such a model of a task for shutting a valve is shown in figure 3.1.

Two dimensions are evident in such a procedural net model. The vertical dimension represents the degree of abstraction. Lower nodes are always expansions of higher nodes. Nodes at the bottom are machine level instructions, nodes at the top are generally process level instructions. The second dimension shows the temporal relationship between nodes. For a given level of abstraction, nodes are related temporally from left to right.

In this case, the node "SHUT VALVE" is an example of a process level description, while all the nodes below it represent machine level descriptions. The arrows show the sequence in which the low level machine commands are executed.

The two dimensional nature of the procedural net model is absent from other possible representations. For instance, flow charts or PERT charts describe temporal relationships, but lack the ability to show how high level concepts are decomposed.

The Warnier-Orr diagram is a representation that includes all features of Perceptronics procedural net model, but also allows decisions to be included. This methodology is based on a set-theoretic system for the logical construction of computer programs. The SHUT VALVE task is shown

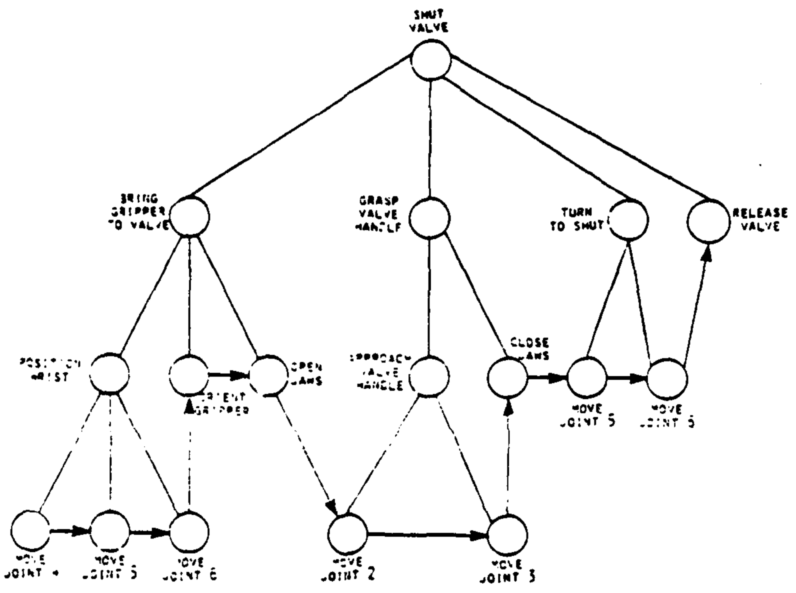


Figure 3.1 Procedural net model for "SHUT VALVE" task (from Shaket et. al.)

in figure 3.2. This diagram resembles the the procedural net model, but the level of abstraction from process level to machine level goes from left to right, and time goes from top to bottom.

Warnier-Orr diagrams can display control structures such as repetition (such as DO...LOOP or DO...WHILE), or alternation (i.e. IF...THEN...ELSE or CASE). This representation is gaining popularity for representing a variety of programs.

3.5 Descriptions for machine level control

Machine level control requires the human operator to communicate in terms of the state of the machine. Rather than telling the machine what he wants done, the operator must describe a time history of machine states that will carry out his intentions. Various methods for implementing machine level control in terms of position and velocity have been described in chapter 2. This section will summarize methods for giving machine level descriptions.

3.5.1 Analogic descriptions

A variety of methods have been employed to allow the human operator to describe movements of the manipulator arm.

The simplest method for controlling a manipulator requires that the operator describe movements of the arm in terms of movements of the

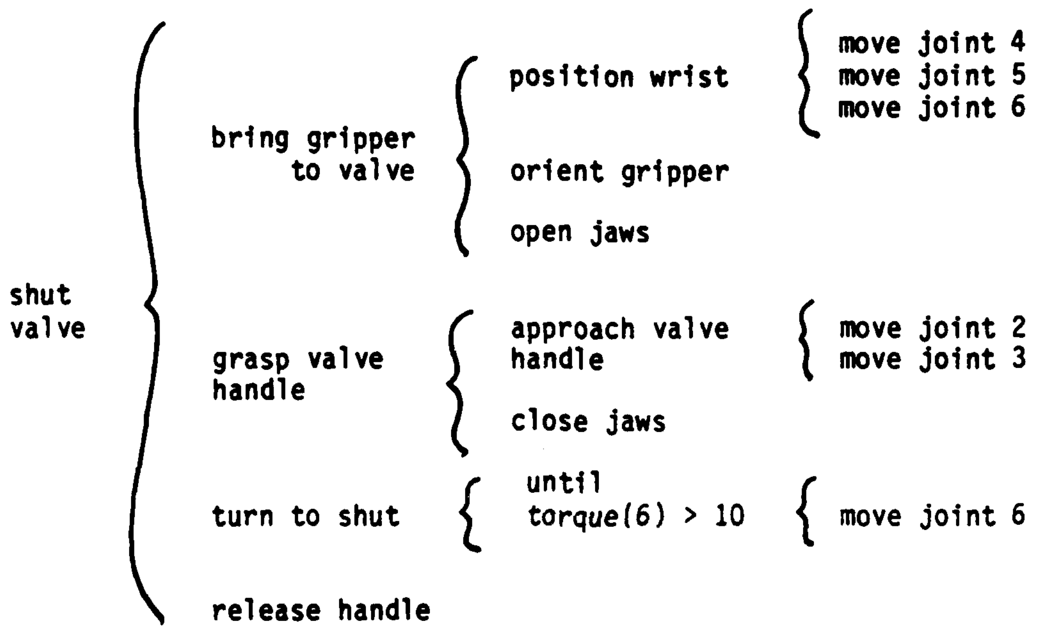


figure 3.2 Warnier-Orr Diagram of Shut Valve task. This is similar to the procedural net model representation, but decisions may also included. In this example, the valve will continue to be turned until a preset value of torque is reached on the last manipulator joint

manipulator joints. Since joint space is seldom convenient for describing tasks, this puts a great burden on the human operator to understand the kinematics of the manipulator arm.

Nearly all remote manipulators are designed with rotational joints, which make this problem particularly difficult. Nevertheless, this is the most common method for control in commercially available underwater manipulators [2]. The motivation for using this control scheme comes from the need for simplicity. Joint actuator controls require no sensing of the manipulator joint positions or velocities; control of the flow of hydraulic fluid or electric current is sufficient. Since measuring the position of articulated joints in the underwater environment makes the system more complicated, this method of control is the most popular.

If the position of the remote manipulator can be sensed, a control system can be used to transform from a space chosen for the human operator's convenience to joint space. In master-slave control, a transformation is done mechanically which allows the human operator to control the manipulator in terms of the position of his own arm. In resolved motion rate control [7], a computer is used to transform cartesian velocities in a convenient coordinate frame to the joint space of the manipulator.

3.5.2 Symbolic movements

Movements can also be stated symbolically. In the case of translation, movement relative to an arbitrary reference can be stated easily in terms of a three dimensional vector.

Rotations may also be stated as a magnitude and a direction. However, rotations do not follow all the rules of vector algebra. In particular, rotations are not commutative. There have been a variety of approaches to this problem.

Rotations are most commonly described by rotation matrices, consisting of direction cosines of one coordinate system relative to another with common origins. This representation is highly redundant, using 9 quantities to represent a 3 degree of freedom rotation. Rotation matrices have the advantage that they can be combined by matrix multiplication.

Another common method for representing general rotation is by a set of simple ordered rotations. These include Euler angles [49], and pitch-roll-yaw coordinates. These representations have some physical meaning to people, but are severely limited by the fact that they may not be combined directly. Commonly, they are converted to rotation matrices, then combined.

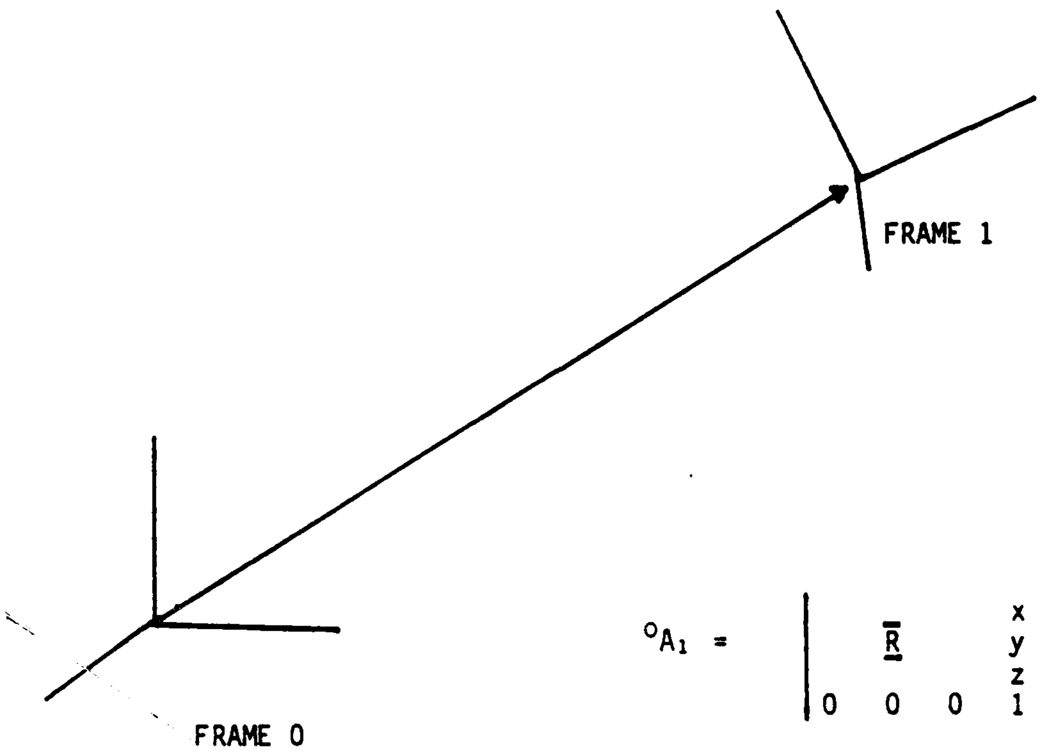
Euler is credited with first proving that any pure rotation can be stated as a single rotation about some axis [50]. This idea was developed by Rodrigues, using a construct known as Rodrigues' coordinates [50]. A more elegant description representing a rotation as a combination of a scalar and a vector quantity is the quaternion, developed by Hamilton [51].

Quaternion methods have been used to represent manipulator rotations by Taylor [26], and Luh [32].

A problem which has proven very difficult is the representation of simultaneous rotation and translation.

The most commonly used method for describing both rotation and translation has been the homogeneous transformation, described by Uicker [25]. This representation combines a rotation matrix with a translation vector to form a 4x4 matrix. Homogeneous transformations were first used for manipulator control by Pieper [21]. This representation is highly redundant, using 16 elements to represent a six degree of freedom quantity. A great advantage of this representation is that the rules of matrix algebra may be applied. A common representation of the homogeneous transformation is shown in figure 3.3.

Numerous unified representations for translation and rotation were studied in the late nineteenth century. These include biquarternions [52], octonians [53], and novenions or nonions. Similar methods, the dual number quarternion, have been used to analyze spatial mechanisms by



$$ {}^{0}A_1 = \begin{vmatrix} & & & x \\ & \underline{\overline{R}} & & y \\ & & & z \\ 0 & 0 & 0 & 1 \end{vmatrix} $$

Figure 3.3 A homogeneous transform describes the spatial relationship between two coordinate frames. The transformation consists of a rotation submatrix $\underline{\overline{R}}$ which describes the orientation of the second frame relative to the first frame, and the translation of the origin of the second frame defined in the first frame.

Yang [54]. These methods have their own algebras, but are not commonly used to represent manipulator motion.

3.6 Descriptions for process level control

3.6.1 Describing tasks by movements

Brooks [10] has shown that models of tasks can be built if a structure is provided for modelling movements of the manipulator arm. Tasks such as opening valves, taking off nuts, and picking and placing operations can be programmed in this way. There were no models of objects in Brooks' system; models of movements were sufficient.

Brooks categorizes positionally defined manipulation tasks into those that can be described relative to the manipulator base (absolute tasks), and those which must be described relative to an arbitrary reference (relative tasks). Additionally, either absolute or relative tasks may be moving or fixed. Brooks presents a mathematical treatment of these four possibilities.

Brooks designed a language which allowed both absolute and relative tasks to be programmed. Relative and absolute motions could be specified analogically and combined with sensing and branching to implement tasks.

Perceptronics [55] also developed a movement based system for teleoperation. Motions could be programmed either symbolically or analogically. Both paths and discrete positions could be recorded. An extensibility mechanism called "chaining" allowed previously defined motions to be combined to form higher level entities.

These studies confirm the value of symbolic and analogic movement descriptions, the importance of both relative and absolute definitions, and the usefulness of extensibility. In both cases, these concepts were demonstrated to be effective through controlled experiments with trained operators.

3.6.2 Describing tasks by relationships between objects

Systems for modelling the environment may be characterized by two features. First, they contain a scheme to represent knowledge of objects in the environment, sometimes including the manipulator. Second, such languages have high level procedures for affecting change in the world as it is represented. Examples of these systems include AL [56], PAL [57], and RAPT [58]. These systems differ from explicit movement control languages, which contain only a scheme for representing movements of the manipulator arm.

All such systems represent items of interest in a tree structure. The nodes of such a tree corresponds to a coordinate frame, and the links represent relative transformations. Transformations need not be static. This representation is very general, and includes Brooks' fixed-moving relative-absolute categorization as a subset, although in this case objects are being represented, not movements.

Three types of models are shown in figure 3.5

RAPT

A body is defined in RAPT by specifying all of the features of that body [58]. Features are defined in terms of a coordinate system imbedded in the body. Features include faces, shafts, and holes. Lower level constructs used to define the features are points, lines, and circles.

Situations are described by stating spatial relationships between features of different bodies and whether bodies are tied to each other. Of particular interest are the descriptions used to represent special spatial relationships. If two faces are stated to be COPLANAR, this means that the two faces lie in the same plane, with their surface normals facing in the same direction. A face may be AGAINST another face, which means that the two faces are coplanar, but their surface normals are opposing. Shafts may also be AGAINST a face of another body. The programmer may state that a shaft FITS a hole (both features are coaxial with normals opposed), or they may be ALIGNED (features are coaxial, normals in the same direction).

Actions define movements of bodies, either in translation (MOVE) or rotation (TURN). Actions are significantly higher level than

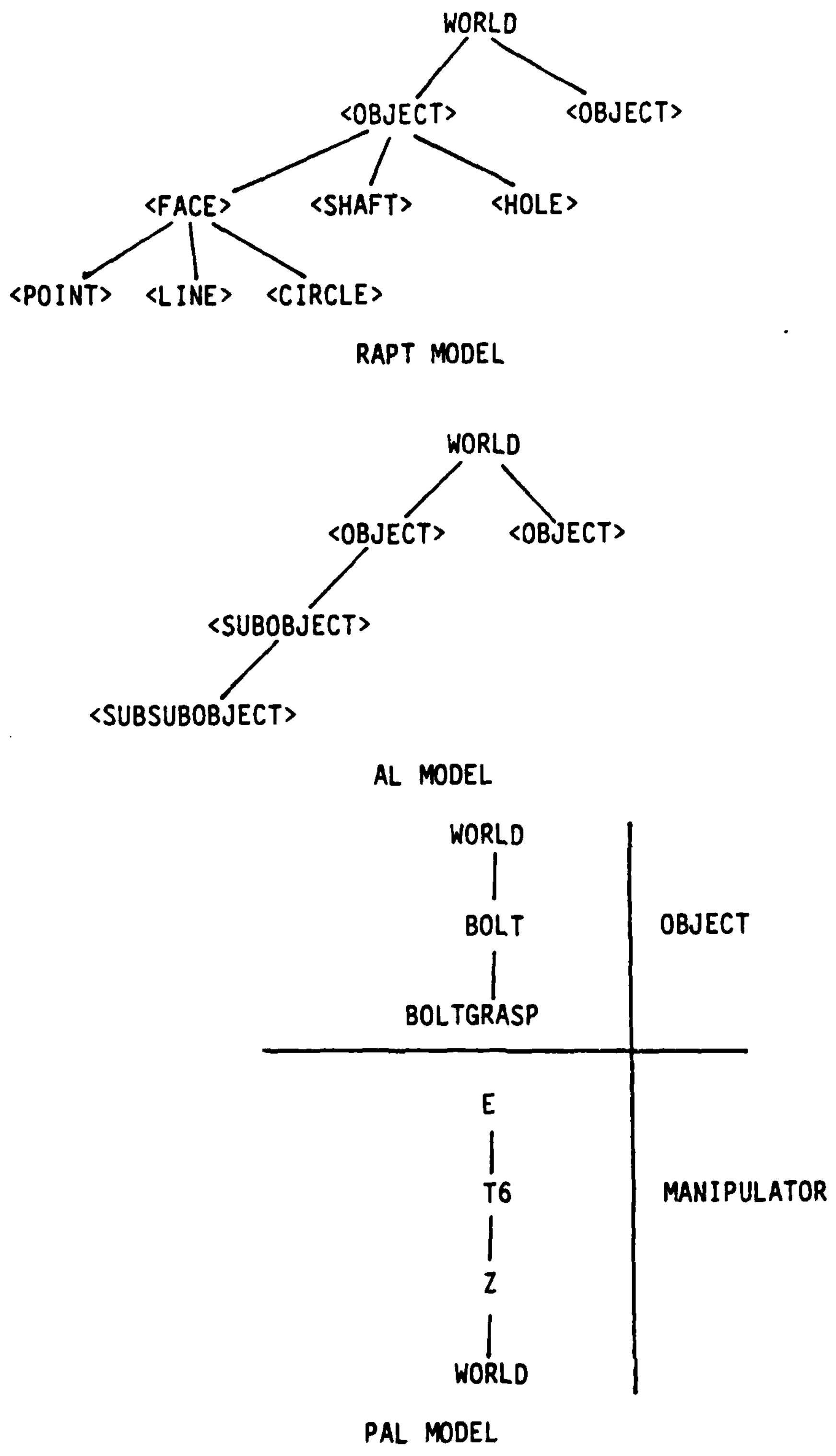


Figure 3.4 World Models

manipulator movements in an explicit language. If a body is MOVEd, all other bodies TIED to that body are MOVEd, both physically and in the RAPT world model. MOVEs may contain explicit directions and distances, or the spatial components of the movement may be infered by first describing the desired situation which the MOVE should produce.

AL

AL was developed at Stanford as a high level language to perform industrial assembly operations [59].

The BODY-FEATURE formalism of RAPT is not used, but a link does imply the lower level object is a subpart or subfeature of the object above it. Also, links can be specified as rigid, non-rigid, or independent. This is more general than the BODY-FEATURE concept in that the tree can be extended to any level, but no size or shape information is present. Relative transforms through which the arm should approach and depart an object are also included.

Movements are specified in AL by first moving the arm to an object, grasping the object, and declaring that the object is affixed to the arm. Now, a movement may be specified in terms of any subobject of the object grasped and any other defined object. After the movement, all objects which have been declared to be affixed to the object moved are also updated.

AL investigators have also incorporated a system for building the model, called POINTY. The structure of the world model may be built and modified with symbolic commands. The arm is used as a measuring device to establish the transformations which relate objects to each other and the world.

Using POINTY, three manipulator positions are used to define an object (actually the coordinate system imbedded in the object is defined). One position and orientation of the arm would be sufficient to establish a coordinate frame. However, it is difficult to orient the hand exactly with the intended frame. Orientation errors are particularly important, since they are greatly magnified during relative movements in the new coordinate system. Using three positions, this orientation error is greatly reduced. Positions can be indicated using the manipulator or a special pointing tool.

PAL

Purdue Arm Language features a world model which also includes the manipulator arm [57].

This model includes both the objects in the environment, and the arm. Links connecting the objects contain relative spatial information.

PAL movements are programmed by entering position equations. Motion statements in PAL request that the manipulator be translated and oriented such that a position equation is satisfied. These position equations represent closed kinematic chains. This approach allows the system to make use of information from a variety of sources, including analogic teaching and data bases.

Position equations may satisfy any of three coincidence conditions between the object moved and the destination frame. These are called = , ^ , and * .

3.7 Summary

This chapter reviews methods which have been designed to allow human operators to communicate with manipulator systems. The distinction is drawn between machine level and process level control, and techniques for achieving both have been described.

Chapter 4

DESIGN OF A MAN-MACHINE INTERFACE FOR SUPERVISORY TELEMANIPULATION

4.1 Summary of System Design Criteria

The supervisory roles of the human operator, planning, teaching, monitoring, intervention, and learning have been described in Chapter 1. This chapter presents an interface that attempts to implement these concepts for application in telemanipulation. The system is called MMIT (Man-Machine Interface for Telemanipulation).

A primary criterion is that the system be designed so that it can be operated by people who do not have a mathematical understanding of manipulator control. While mathematically oriented interfaces are very general and powerful, such interfaces require that the operator understand manipulator control in precise mathematical terms, rather than in intuitive terms.

The system has been designed to allow the human operator to participate as fully as possible during teaching. The operator teaches the system how to execute tasks using a movement control language. This language has many features of a modern programming language.

Features of the language include specialized data structures for teaching positions and orientations analogically. Movements can then be programmed by combining analogic position data with symbolic movement commands. Decisions can be included using structured flow of control statements. This allows programs to be written where the operator points out an object in the environment (particularly a weld or a surface), then describes an operation on that object (such as an inspection task) by describing motions built on the coordinate frames that the have been pointed out.

Manipulator tasks can be composed as a hierarchy of subtasks. Modular design of manipulation procedures is encouraged through a general extensibility mechanism. A text editor and a file system are also included.

The operator can request that the system demonstrate how the operator's high level instructions have been implemented. Using computer graphics, commands can be simulated before any movement of the manipulator occurs. The concept of a computer control system being "apparent" [1] is an important goal.

The human operator remains in control of the system during execution. The operator can use the graphic display to aid in monitoring the execution of a task. Tasks can be interrupted if the human operator detects a problem which the system cannot detect. The operator can take over manually, or he can invoke computerized functions to get out of trouble.

4.2 Mathematical Movement Control Language

The human oriented movement control language is implemented in a more general, mathematical language. The human operator does not see the mathematical interface, but this language is used to implement higher level languages. The language includes specialized data types pertaining to manipulation, routines for kinematic computations, transformations, sensing, and control of motion. This language is summarized in Appendix D.

4.3 Interactive Computer Graphic Display

An important element of the system is a dynamic computer graphic display (figure 4.1). The display was designed by Winey [60], and developed further by Fyler [61]. The display allows the operator to view the image of the environment from any viewpoint. The operator may rotate, translate, or zoom the manipulator display. Several examples are shown in figure 4.2. The manipulator display is controlled from a touch panel interface designed by Washington [62] (figure 4.3) or under program control.

The display has three major functions:

1. The display is a very effective monitoring aid, allowing the operator to view the arm and representations of the environment from any desired viewpoint.

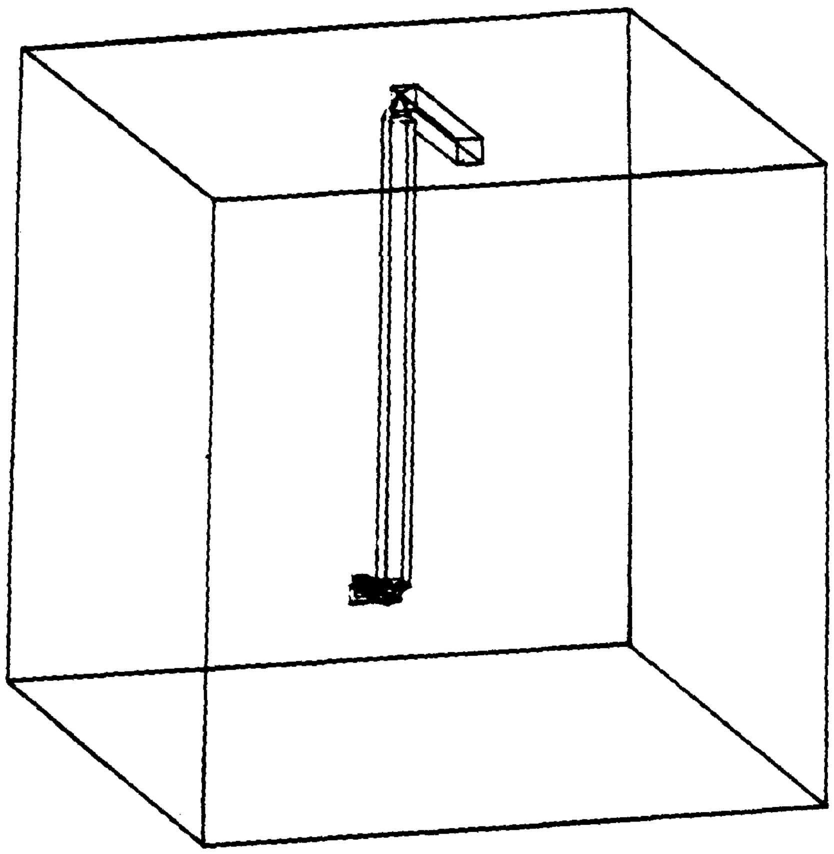


Figure 4.1 Computer graphic display, standard view.

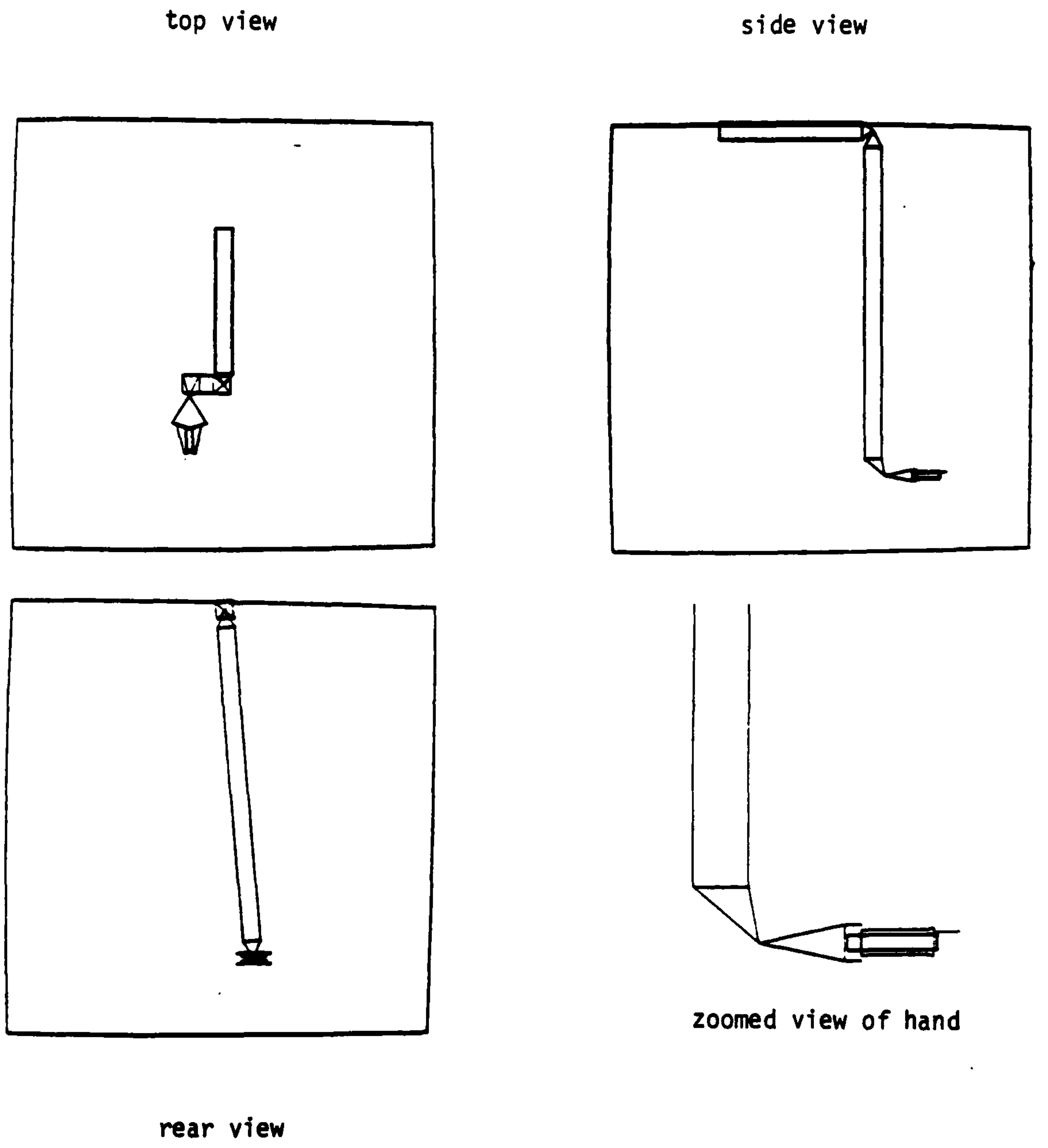


Figure 4.2 The graphic manipulator may be viewed from and direction, zoomed, or translated.

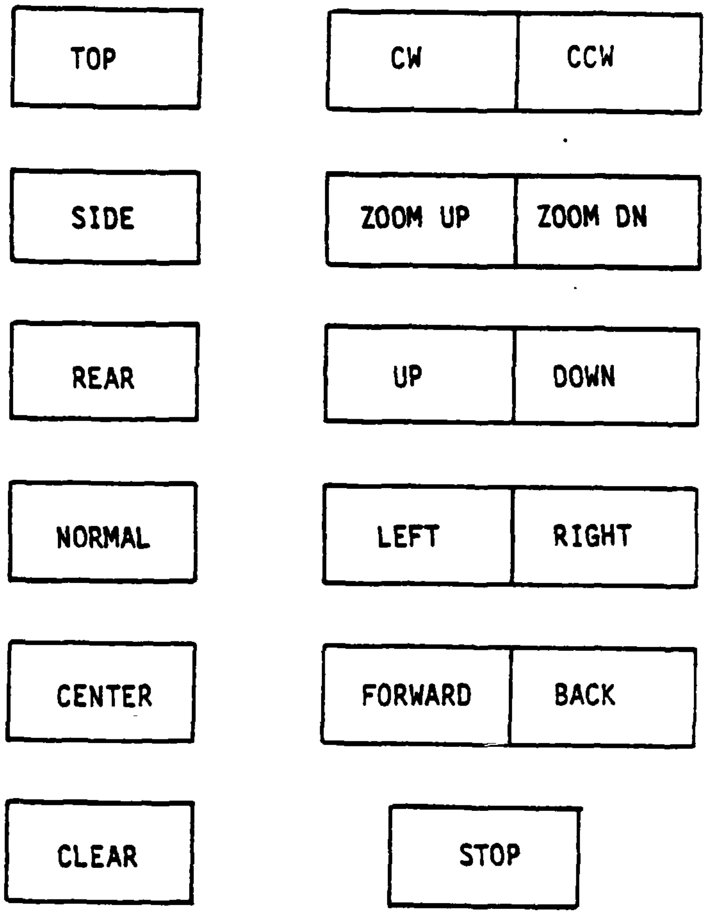


Figure 4.3 Touch panel display used to control manipulator display

2. The display can be used to test many programs before the programs are run with the real arm. Simulations may be run with time speeded up.

3. The display can be used to show the result of all computation done by the system after high level descriptions have been given. This helps people to understand how the system works.

4.4 Fundamental Definitions

A formalism is needed to precisely specify a language. Computer languages are commonly specified using Backus-Naur Form (BNF) [63]. In BNF, an entity is specified in terms of other defined entities and possibly itself. BNF descriptions are an unambiguous way of specifying the syntax of a language.

The complete formal definition of the language in BNF appears in Appendix E. Appendix E contains sufficient detail for a user to learn to use the system. The treatment in this chapter will be more descriptive, although less precise.

The formal definition of the language begins with the definition of integers and names. An integer is any 16 bit number (between -32768 and 32767). Integers will be used to specify distances, angles, and as indices. A name may consist of any string of characters, up to 31 characters long. Names may contain any character on the keyboard except for space. Since coordinate frames are defined analogically, they are given names that allow these frames to be referred to later.

Examples:

STARTING_POSITION

TWIST

4.5 Structures for Defining Coordinate Frames Analogically

One method of programming the manipulator will be to first define a number of positions and orientations analogically (positions and orientations will be referred to simply as positions for the remainder of this section). Motions are not taught analogically, just positions. A position may be taught absolutely (fixed relative to the base). Relative positions, which are defined in relation to any specified reference, may also be taught. These positions can then be combined in a program with symbolic commands to produce motions. All coordinate frames defined in this way may be shown on the graphic display.

Analogic definitions allow the operator to establish positions quickly. The accuracy of this process under remote viewing will be examined later in Chapter 6. Analogic definitions are especially useful when the operator has no way of knowing he exact values of the coordinates of a position, but he can see what he wants. Using analogic definitions, he may define these positions by moving the arm.

Analogically defined absolute positions are useful for establishing the locations of tools, paths to follow, and especially for teaching the system about objects. This process will be examined experimentally in Chapter 7.

Relative positions can be defined analogically and combined with symbolic commands to define motions which can be executed anywhere in the workspace of the arm. The motion required to turn a valve or a weave pattern to be used in an inspection task can be easily taught in this fashion.

4.5.1 Analogic control modes

The Argonne E2 manipulator is controlled in a master-slave mode [10] or in resolved motion rate control (RMRC) using a 3 dimensional force joystick designed by Griesser [64]. The NOSC manipulator is controlled manually with RMRC. The implementation of RMRC solves the inverse kinematics at approximately 10 hz., rather than using an inverse jacobian technique discussed in Chapter 2.

4.5.2 Absolute coordinate frames

An absolute coordinate frame is defined relative to the fixed base coordinate system. This relationship is shown in figure 4.4. The homogeneous transform representing an absolute frame is that transformation which would translate and rotate the base frame into the

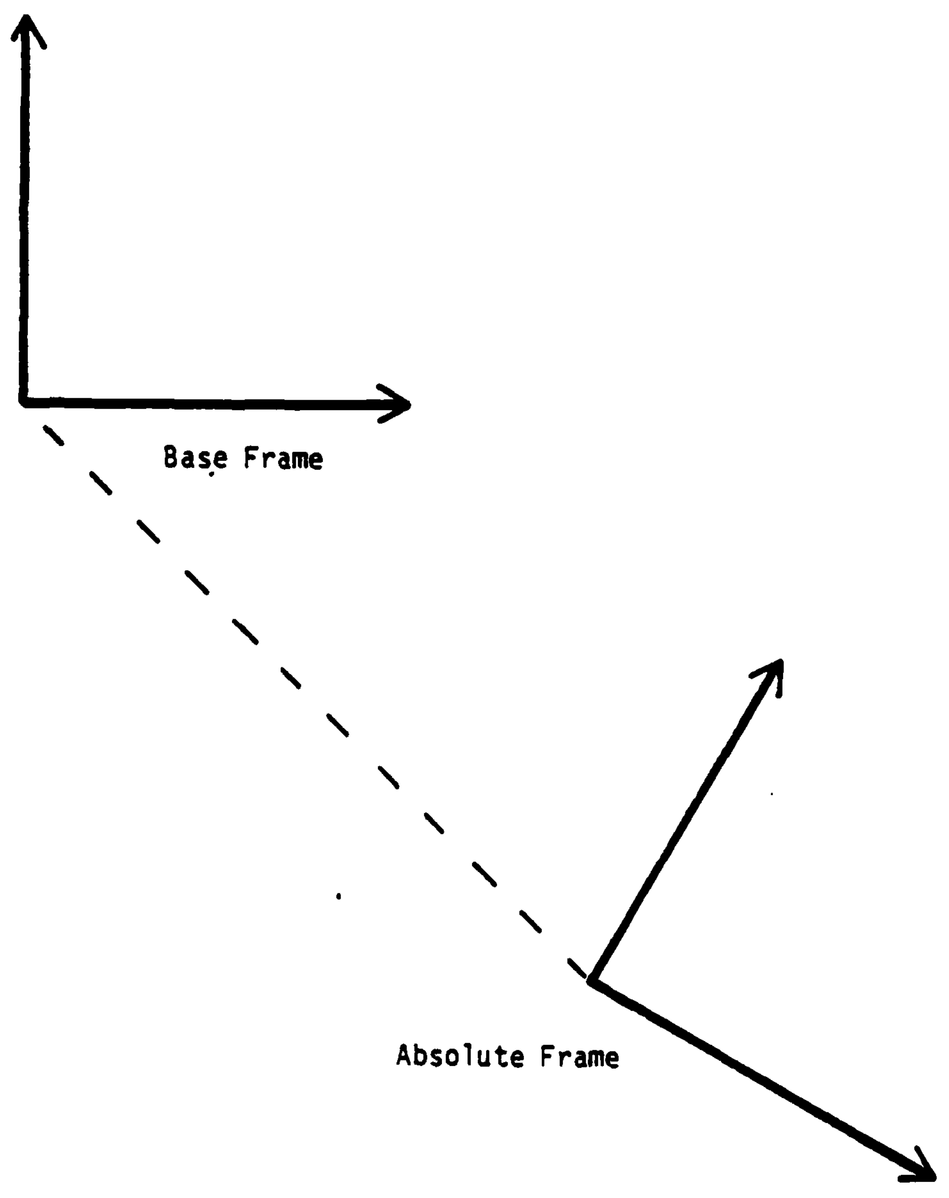


Figure 4.4 An absolute coordinate frame. Absolute coordinate frames are defined relative to the base frame, as indicated by the dashed line.

absolute frame. A special absolute coordinate frame is the hand frame, which moves with the hand. This frame may be defined in Brooks' (as shown in Chapter 2) notation as:

A

Several structures have been designed to define absolute coordinate frames, so they may be used to program movements.

The first of these structures is called a POSITION. A POSITION is first given a name, then defined by moving the arm to the proper position and orientation. The operator indicates when the desired position and orientation have been reached by pushing a button. When the button is pushed, the cartesian hand frame is computed based on the current angles of the manipulator joints and stored under the given name. A POSITION may be shown on the graphics display with the SHOW command, as shown in figure 4.5.

Any number of absolute frames can be established as a PATH. Each frame in a PATH is entered in the same way as a POSITION, although the final position is indicated with a different button. In addition to the coordinate frames which make up the PATH, the number of frames is also stored for use in programs which use PATHs. Individual transformations of a PATH may also be referenced. PATHs may be shown on the display with the SHOW command. An example of a PATH is shown in figure 4.6.

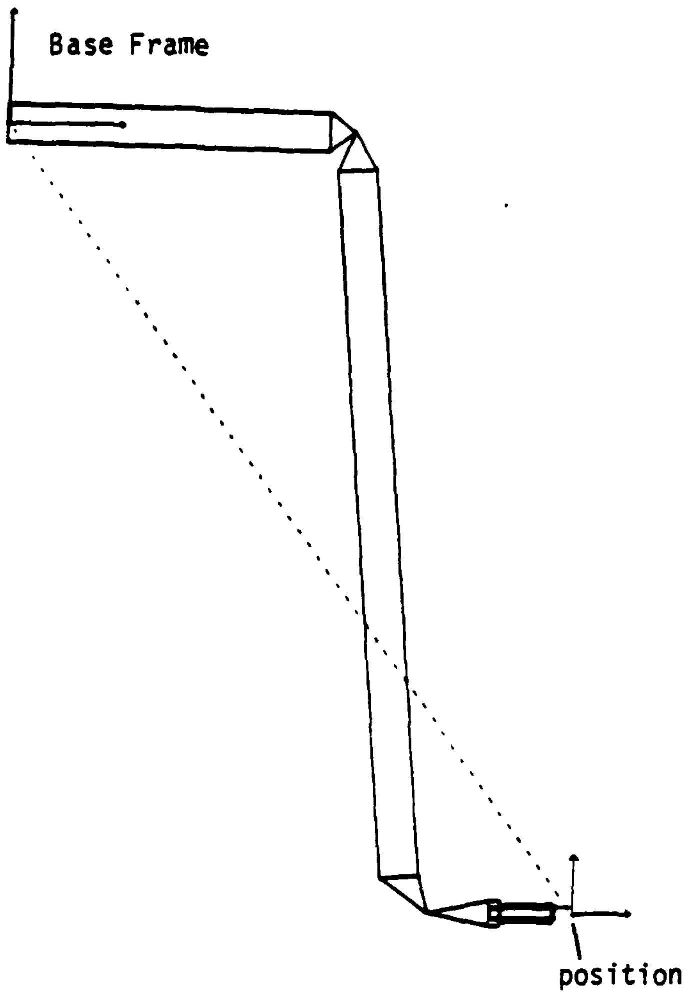


Figure 4.5 Definition of a POSITION. A POSITION may be defined by moving the arm to the desired position and orientation, then recording the result under a descriptive name. A POSITION is described by a homogeneous transformation defined relative to the manipulator base.

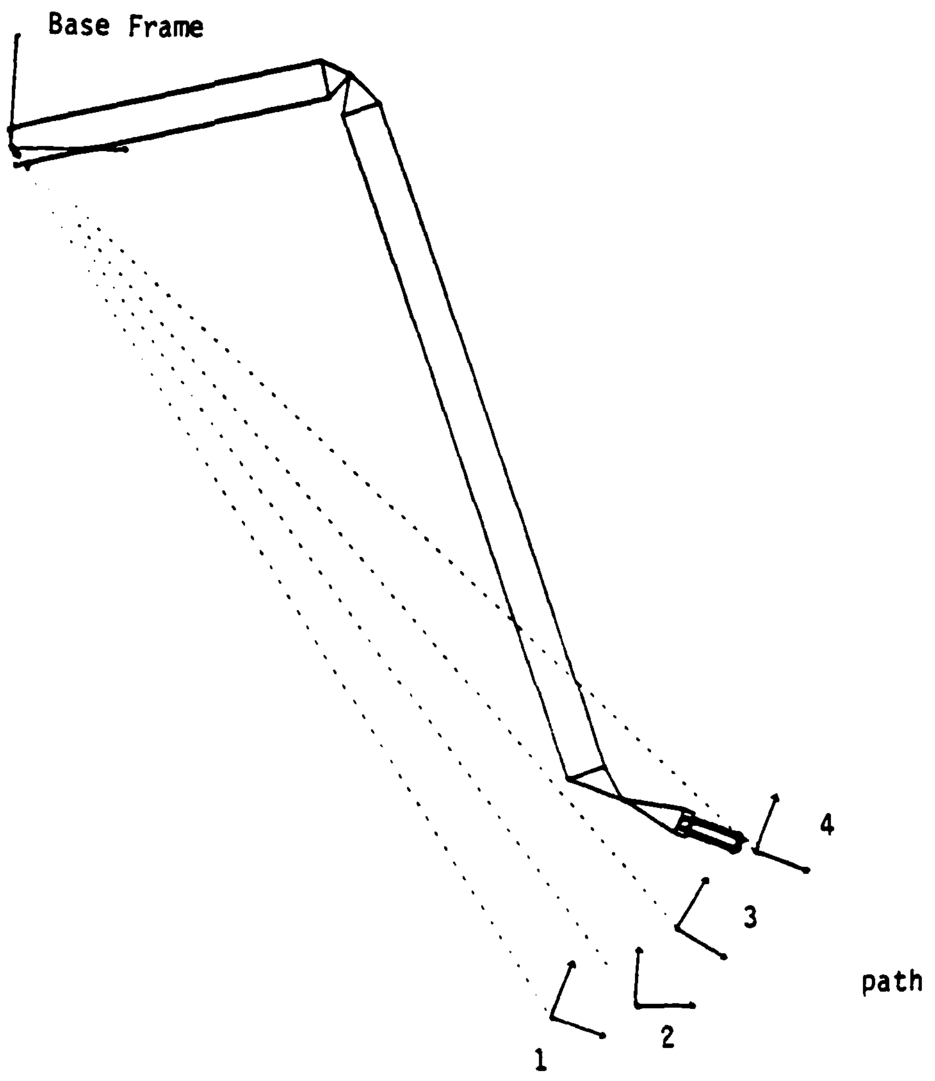


Figure 4.6 Definition of a PATH. A PATH consists of a series of positions and orientations defined absolutely.

4.5.3 Relative coordinate frames

A relative coordinate frame (REL_FRAME) may be defined by by pointing out two absolute coordinate frames with the arm. The system then computes the relative transformation between these two absolute frames and stores the relative transformation under the given name. This procedure is shown in figure 4.7. A REL_FRAME may be applied to the current hand frame and displayed with the SHOW command.

Another structure, REL_PATH, allows a chain of relative transformations to be recorded under a single name. An example is shown in figure 4.8. The SHOW command allows a REL_PATH to be displayed relative to the current hand frame.

4.6 Defining Motions

Motions can be programmed either by using any analogic structures that have been created, or motions can be defined purely symbolically. Any motion can be simulated using the computer graphic display by first issuing the SIMULATE command.

4.6.1 Relative motions

The manipulator hand may be moved to a position and orientation defined relative to the current hand frame using the MOVE command, as shown in

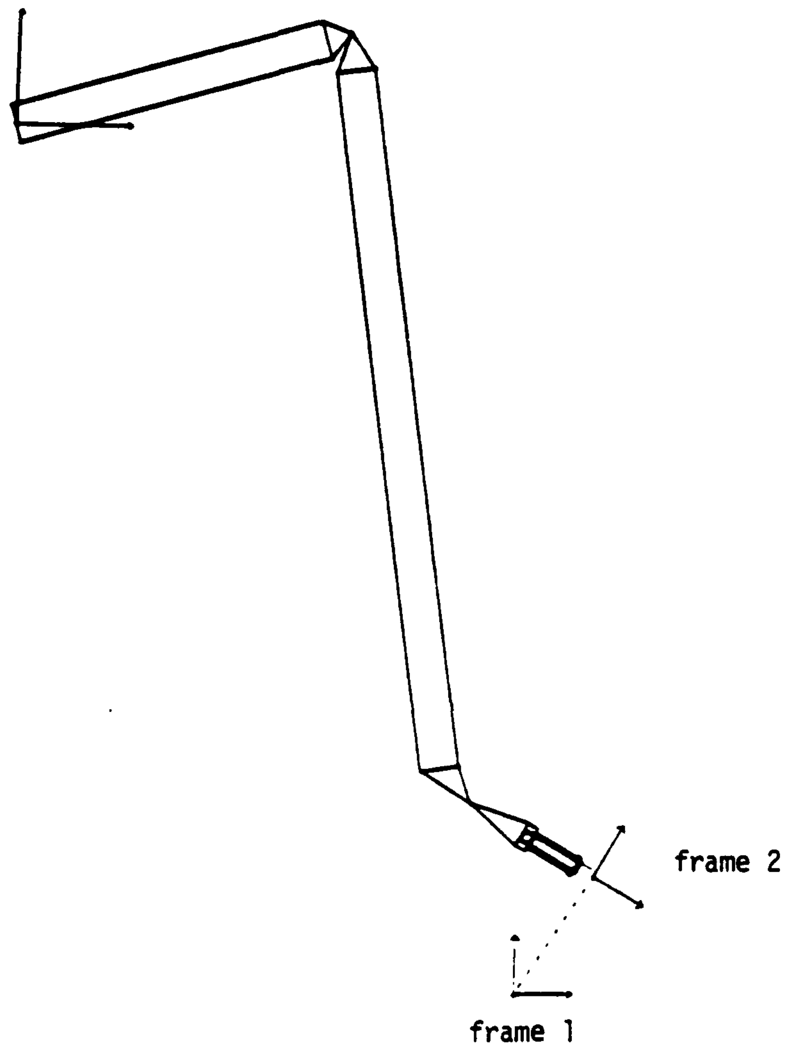


Figure 4.7 Definition of a REL_FRAME. A REL_FRAME is defined by indicating two positions and orientations. The homogeneous transformation which would transform the first frame into the second is recorded under a descriptive name.

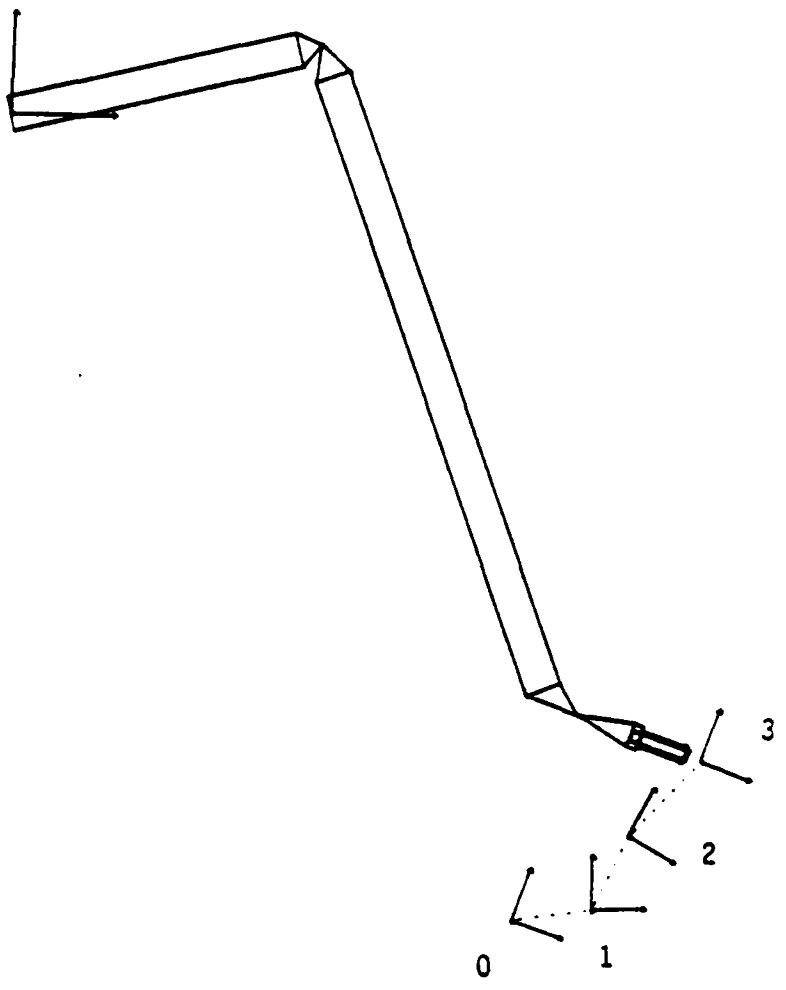


Figure 4.8 Definition of a REL_PATH. A REL_PATH is defined by moving the arm through a series of positions and orientations. Positions and orientations are stored as a chain of relative transformations.

figure 4.9. The MOVE command may be preceded by any properly defined relative frame.

Relative motions may also be defined symbolically. Using symbolic relative motion commands, it is possible to make the arm translate or rotate relative to the current hand frame. The directions UP, DOWN, LEFT, RIGHT, BACK, and FORWARD are defined in the hand frame, as shown in figure 4.10. Symbolic translation along a single axis of the hand frame may be specified by indicating a distance and a direction. A distance is composed of an integer followed by a unit of measure for translation (CM, MM, INCHES, or FEET). A general translation may be specified by three distances and the TRANSLATE command.

Examples:

```
10 CM UP
6 INCHES BACK
0 CM   10 CM   5 CM TRANSLATE
```

The three arguments for the TRANSLATE command are for the RIGHT, FORWARD, and UP (x,y,z) directions. Several other examples are shown in figure 4.11.

Relative rotations may be specified by an integer corresponding to the magnitude in degrees, and an angle, either PITCH, ROLL, or YAW.

Examples

```
45 ROLL
-30 YAW
```

More examples are shown in figure 4.12.

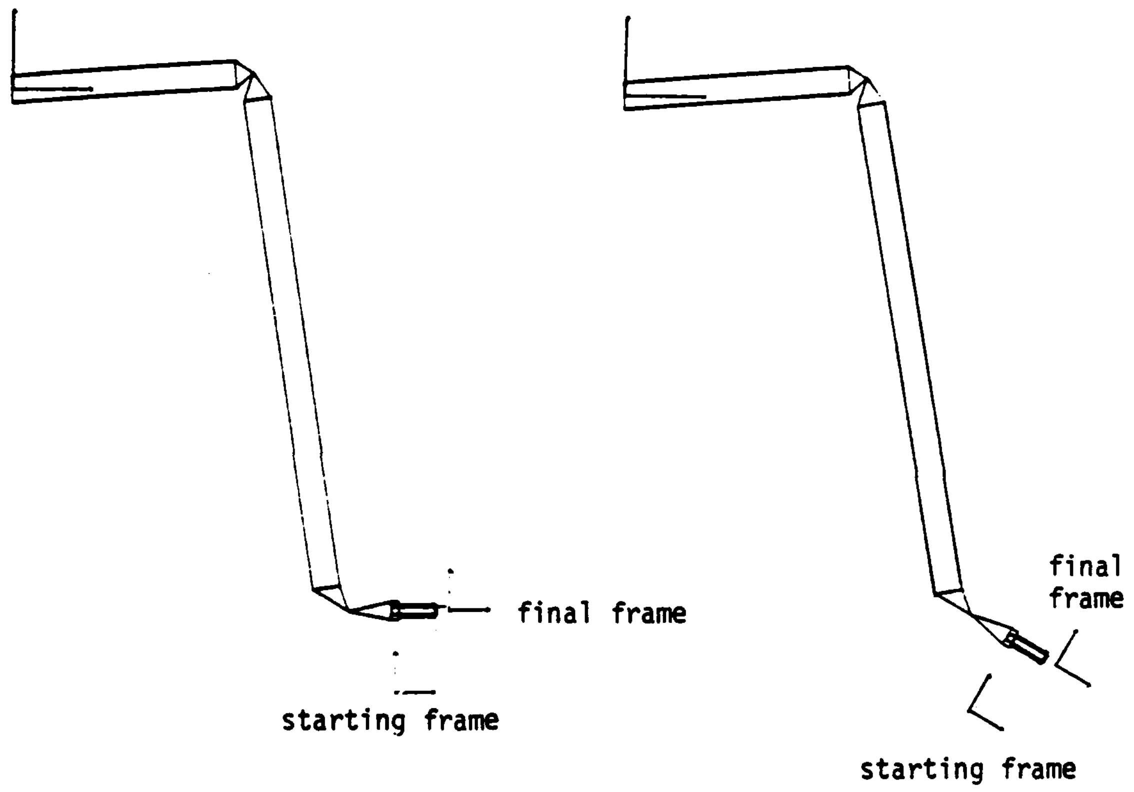


Figure 4.9 A relative motion executed from different starting positions. In both cases, a relative translation of 20 cm in z and 10 cm in y has been executed relative to the starting hand frame.

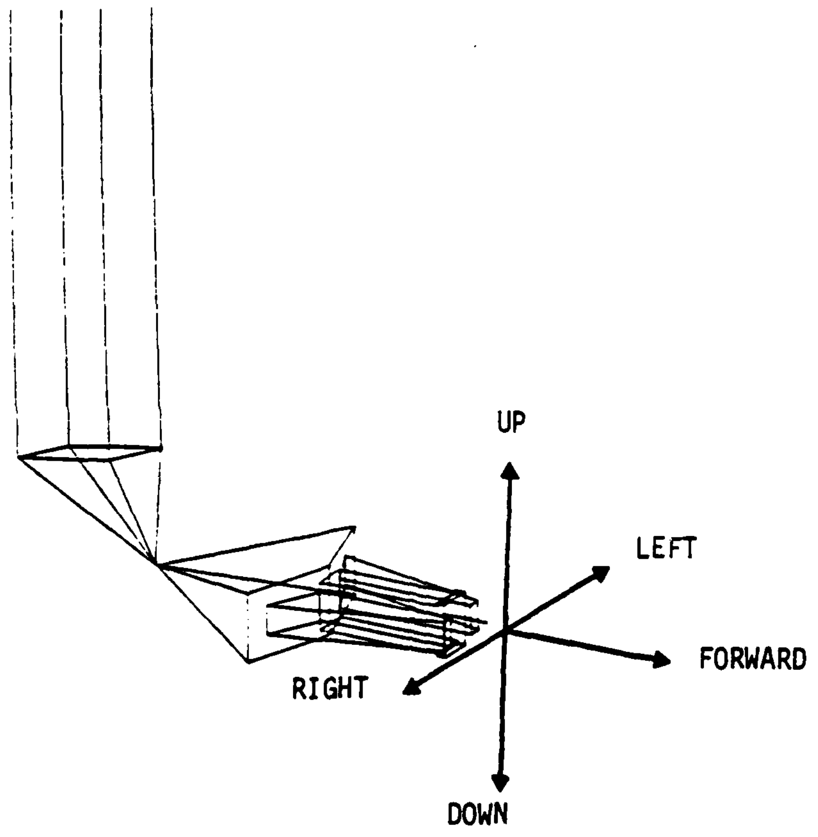
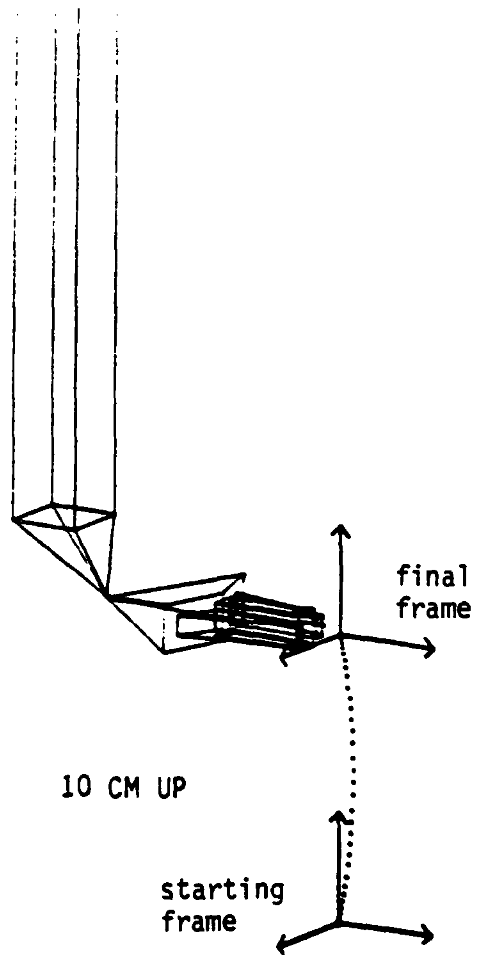


Figure 4.10 Directions for symbolic translations are defined relative to the hand frame.



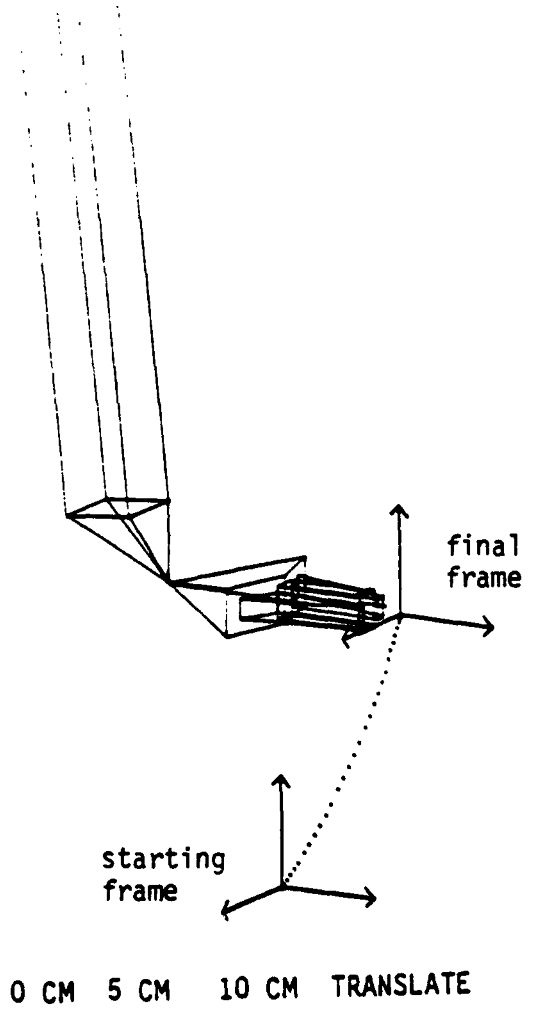


Figure 4.11 Symbolically defined relative translations.

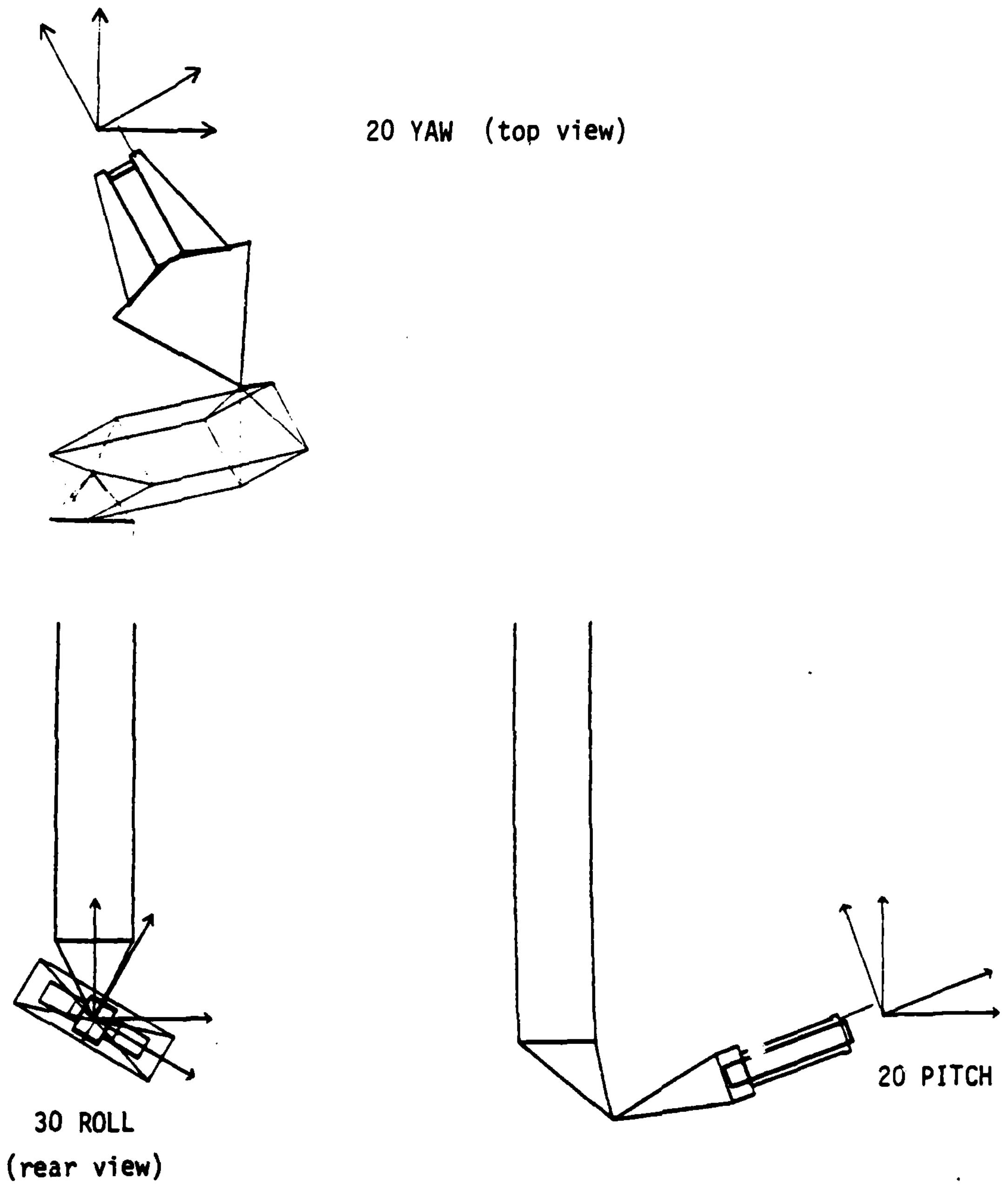
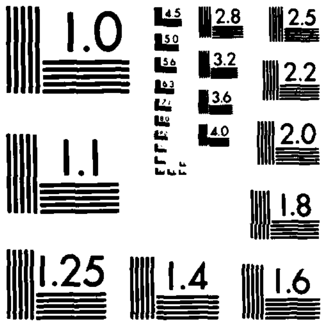


Figure 4.12 Symbolically defined relative rotations.

AD-A123 632 SUPERVISORY CONTROL OF UNDERWATER TELEMANIPULATORS: DESIGN AND EXPERIMENT(U) MASSACHUSETTS INST OF TECH CAMBRIDGE MAN-MACHINE SYSTEMS LAB D R YOERGER 2/3

UNCLASSIFIED 30 AUG 82 N00014-77-C-0256 F/G 5/8 NL


1.0
4.5
5.0
5.6
6.3
7.1
8.0
2.8
3.2
3.6
4.0
2.5
2.2
2.0
1.1
1.8
1.25
1.4
1.6

4.6.2 Absolute motions

Unlike relative motions, absolute motions always take the arm to the same position and orientation. Commands have been designed to move the arm to any absolute coordinate, or directly to a frame defined relative to an absolute coordinate.

The MOVE command may also be used with a POSITION. This command causes the arm to move from its current position and orientation to the coordinate frame defined for the position. An example is shown in figure 4.13.

Another type of absolute motion is used to move the arm to a position or orientation defined relative to an absolute coordinate frame. This command is called SET. Examples:

For a POSITION called POS1

```
POS1 MOVE
POS1 SET 5 INCHES BACK
```

These commands are both illustrated in figure 4.14. Although use of the SET command involves a relative motion, the entire sequence defines an absolute move, since the arm will move to the same position regardless of the starting position.

4.6.3 Relative trajectories

A sequence of relative movements is called a relative trajectory.

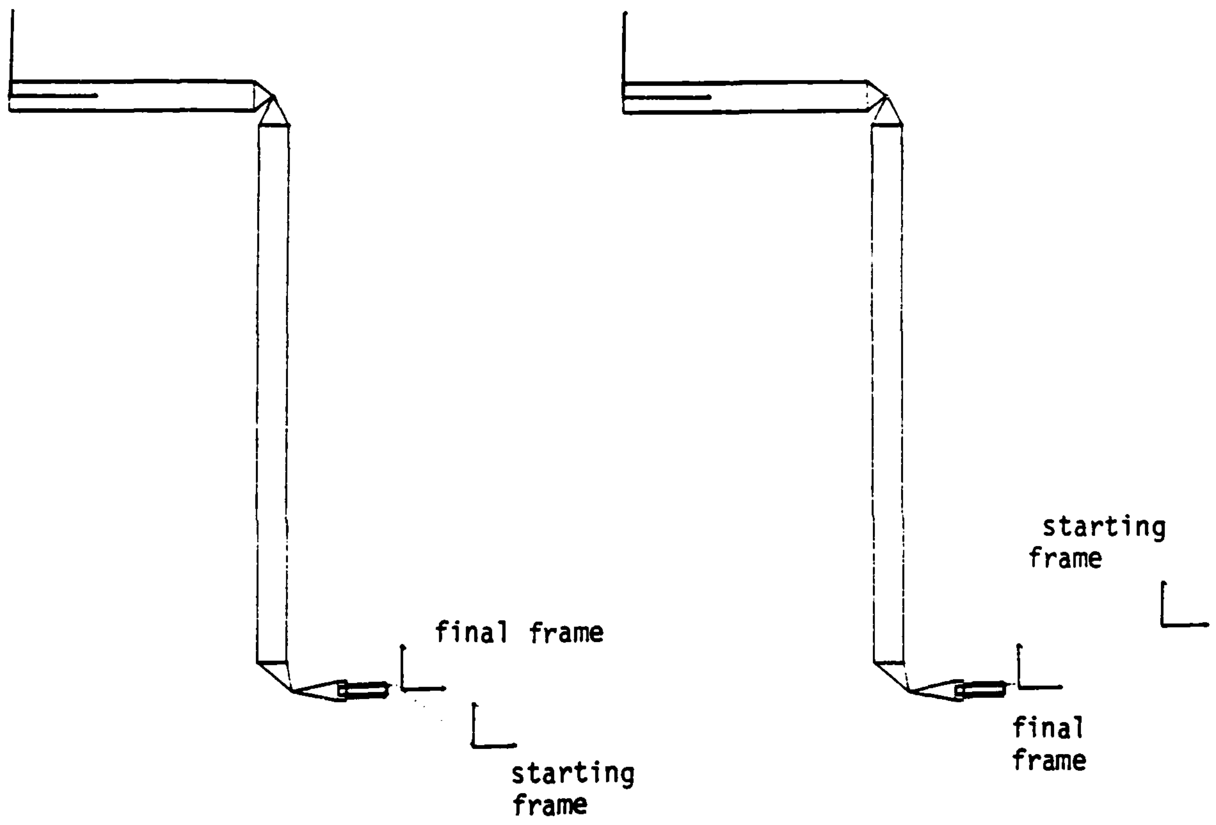
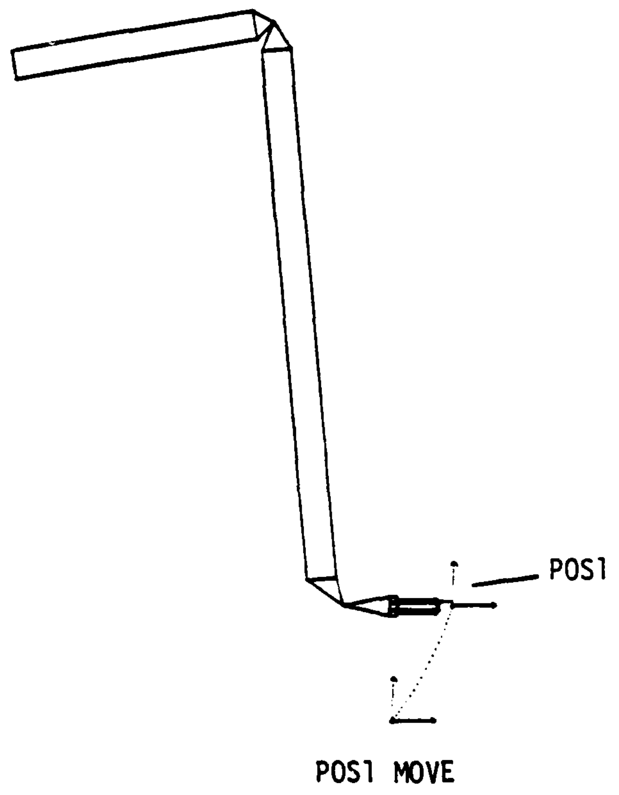


Figure 4.13 An absolute motion executed from different starting positions. The final position is identical regardless of the starting position.



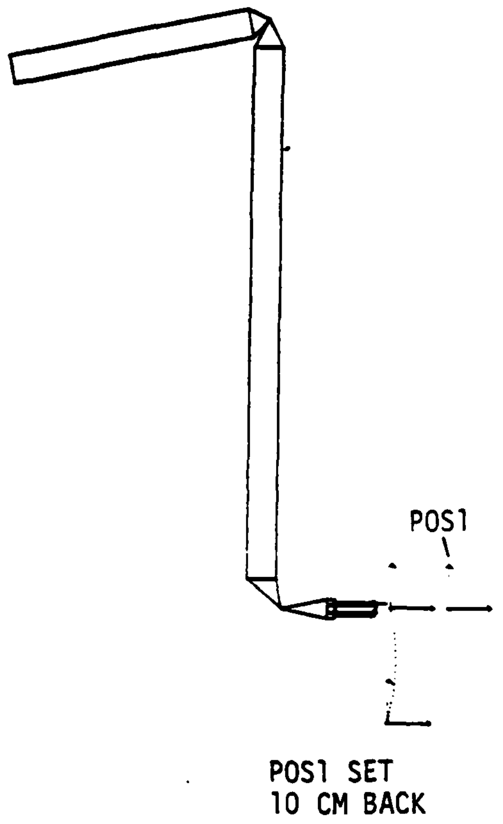


Figure 4.14 MOVE and SET commands for absolute motions.

Relative trajectories, like relative motions, are executed relative to the hand frame when the trajectory is commanded.

Relative trajectories may consist of a series of symbolic translations or rotations. Another useful command is the TRAVERSE command that causes the sequence of relative motions stored as a REL_PATH to be executed. Figure 4.15 shows a relative trajectory executed from two different starting hand frames.

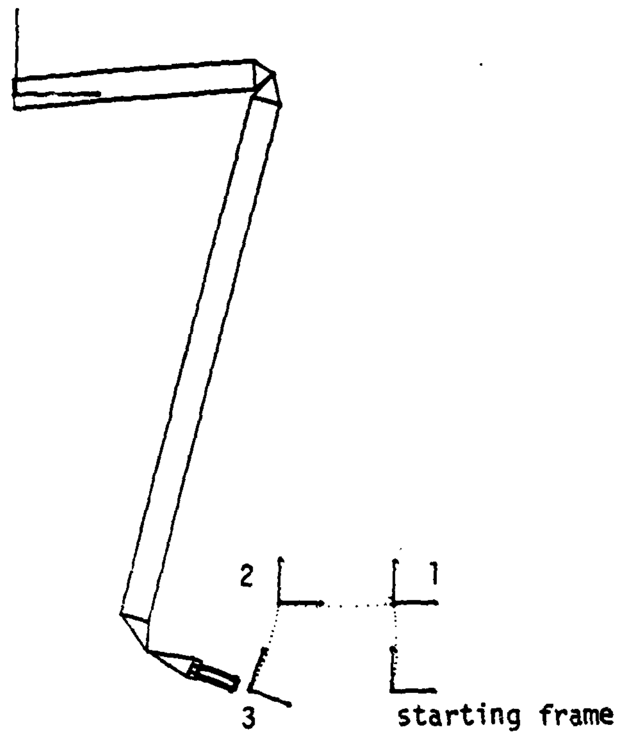
4.6.4 Absolute trajectories

Absolute trajectories are sequences of movements that begin with an absolute movement. A common example of an absolute trajectory is to move through the sequence of absolute coordinate frames stored as a PATH. The TRAVERSE command moves the arm through all frames in a PATH in the order they were defined. The REVERSE command moves the arm through all the frames of a PATH in reverse order. Figure 4.16 illustrates an absolute trajectory executed from two different starting hand frames.

4.7 Sensing Commands

For the Man-Machine Systems Lab version, there are commands for sensing the state of the arm and its sensors.

The first command ?TOUCH returns TRUE or FALSE depending on the state of



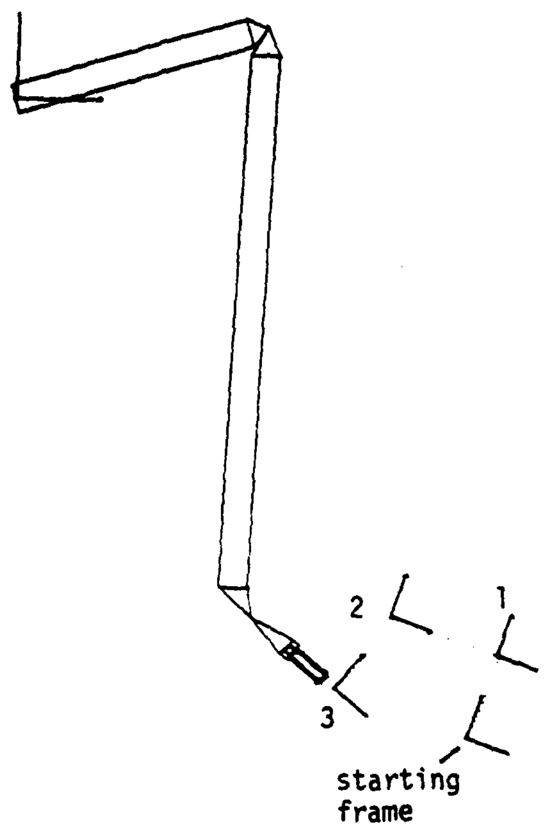


Figure 4.15 A relative trajectory executed from two different starting positions.

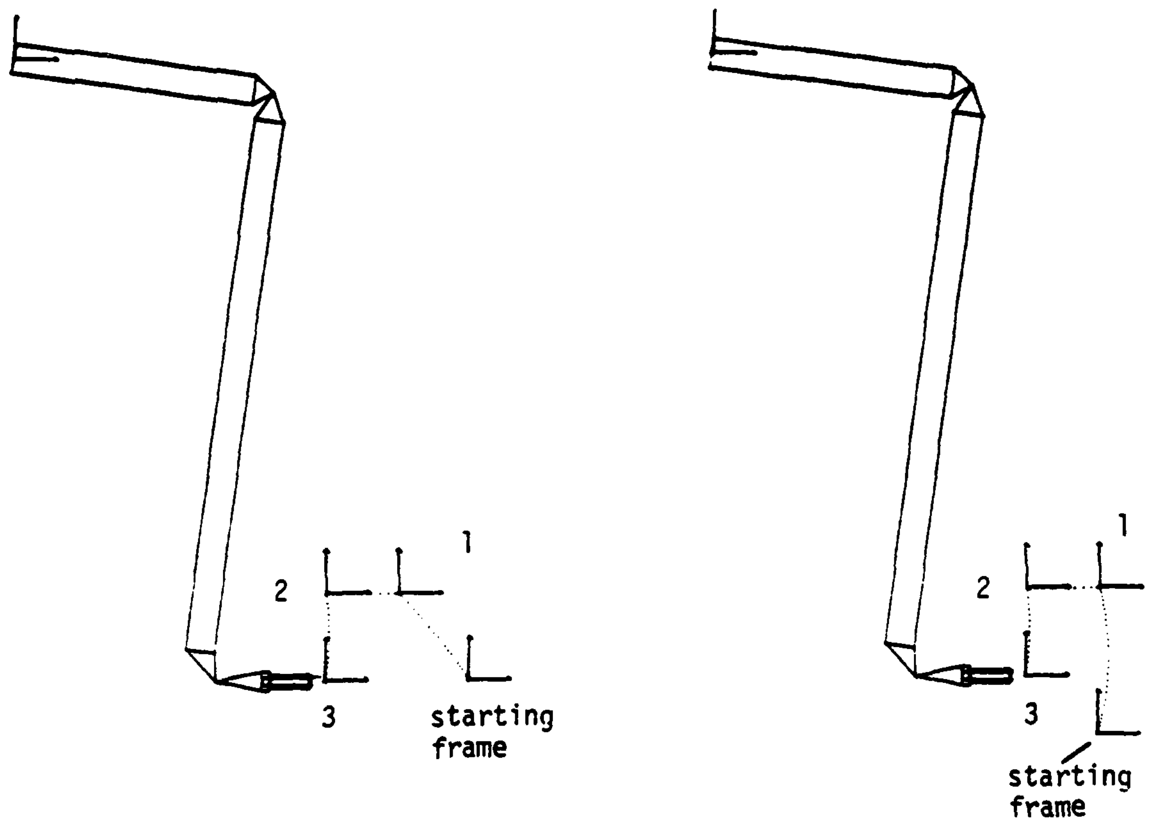


Figure 4.16 An absolute trajectory executed from two different starting positions.

the touch sensor mounted on the manipulator hand. The touch sensor, designed by Fyler [61] can sense contact force in any direction.

Another command ?FORCE returns TRUE if the estimated hand force exceeds a preset value. A similar command ?TORQUE returns TRUE if the resolved hand torques exceed a preset value.

4.8 Building Manipulation Tasks Using Movements

Manipulation tasks may be composed at the machine level by combining motion commands with sensing and branching commands. Sequences of such commands can be combined under a descriptive name to create a new command. This new command may then be used with other commands to create more commands. This allows a heirarchical approach to programming.

4.8.1 Building elements of a hierarchy

When commands are grouped together to create a new command, the new command becomes as much a part of the system as the old commands that were used to create the new command. Programming in such a system may be seen as a process of extending the language to meet a user's needs.

The interpretive portion of the system is implemented in FORTH (the fig-FORTH version [65]), and such extensibility is an essential feature of FORTH. This mechanism is efficient; new definitions are compiled after they are interpreted, so the poor performance of most interpretive systems is avoided.

A new command may be defined starting with the colon symbol, giving a name, then listing any previously defined commands, and ending with a semicolon.

Example:

```
: SQUARE        5 INCHES RIGHT
                5 INCHES UP
                5 INCHES LEFT
                5 INCHES DOWN ;
```

Now the operator can make the manipulator execute this sequence by simply typing "SQUARE", or the operator may use SQUARE in another definition.

For users who are familiar with FORTH, parameters may be passed to commands via the FORTH stack. For those who do not know FORTH, a simple mechanism has been developed for passing single parameters to a command. The command DEFINE: is used in place of the standard colon command.

Example:

```
DEFINE: SQUARE     PARAM RIGHT
                   PARAM UP
                   PARAM LEFT
                   PARAM DOWN ;
```

Now the command SQUARE can be used as follows:

```
5 INCHES SQUARE
 503 MM SQUARE
```

where 5 inches and 503 millimeters are the parameters. A parameter may be a distance, the size of a rotation, or the name of any the analogic structures (POSITION, PATH, etc.).

4.8.2 Repetition and alternation

Programs may be written that branch depending on the sensing commands or that repeat a number of times. Structures for this include DO...LOOP, IF...ELSE...ENDIF, and BEGIN...UNTIL. Each of these structures must appear inside colon or DEFINE: definitions.

An example using such a structure would be a command to close a valve up to a certain torque:

```
: TIGHTEN      BEGIN
               2 ROLL
               ?TORQUE UNTIL ;
```

This procedure may be displayed as a structure diagram:

```
        ( until              (
TIGHTEN { torque limit       {  roll 2 degrees
        ( exceeded           (
```

After the command TIGHTEN is received, the system will attempt to roll the manipulator wrist in 2 degree increments until the preset torque value is exceeded.

The SHUT VALVE example from chapter 3 may also be programmed in this way:

The structure diagram:

```
            ( prompt the operator "move arm to valve"
            (
            ( enter manual control, exit when button is pushed
            (
SHUT_VALVE  { close jaws
            (
            ( tighten valve to torque limit
            (
            ( open jaws
```

Since modules already exist to accomplish all of these steps, the task may now be coded. The corresponding program is:

```
: SHUT_VALVE    PROMPT: MOVE ARM TO VALVE"
                 MANUAL
                 CLOSE
                 TIGHTEN
                 OPEN  ;
```

4.9 Structuring the Environment

Several devices are included to structure information about the environment. These include plane surfaces, general surfaces, and welds.

These entities were chosen as they are useful for underwater inspection operations.

A structure for establishing an absolute coordinate frame corresponding to a flat surface is called a PLANE. A PLANE is made by first moving the arm to a position and orientation in front of a flat surface. The system will then determine the position and orientation of a flat surface in front of the hand by touching the surface in three spots. A PLANE can be referred to by name or shown on the display (figure 4.17). The coordinate transformation so defined can be used like a [illegible]ITION with the MOVE and SET commands.

A more general structure represents a surface as a series of absolute *coordinate frames. This structure* is called a SURFACE. The operator moves the arm to a starting position, as with a PLANE. The system then identifies a preset number of coordinate frames, touching the surface at three points to define each frame.

Welds may also be represented by a series of absolutely defined coordinate frames, called a WELD. The operator indicates an approach frame through which the system should approach the weld, then indicates a series of positions that are touching the weld, and finally indicates a departure frame. On the graphic display, the approach and departure frames are shown as coordinate frames, and the weld is shown as a series of points (figure 4.18).

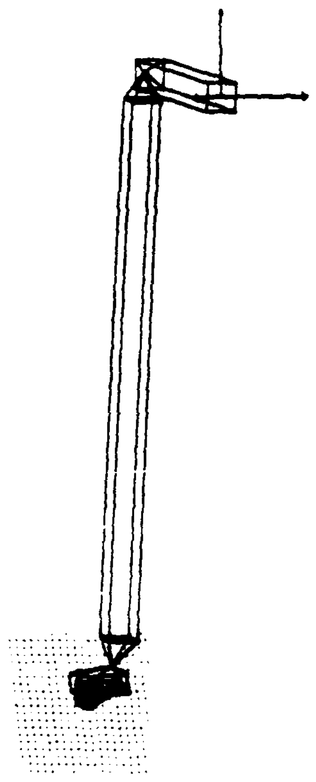


Figure 4.17 A PLANE is defined by the system by touching a surface at three points. A PLANE is stored as a homogeneous transformation defined relative to the base. PLANES may be displayed on the computer graphic display as shown in this figure.

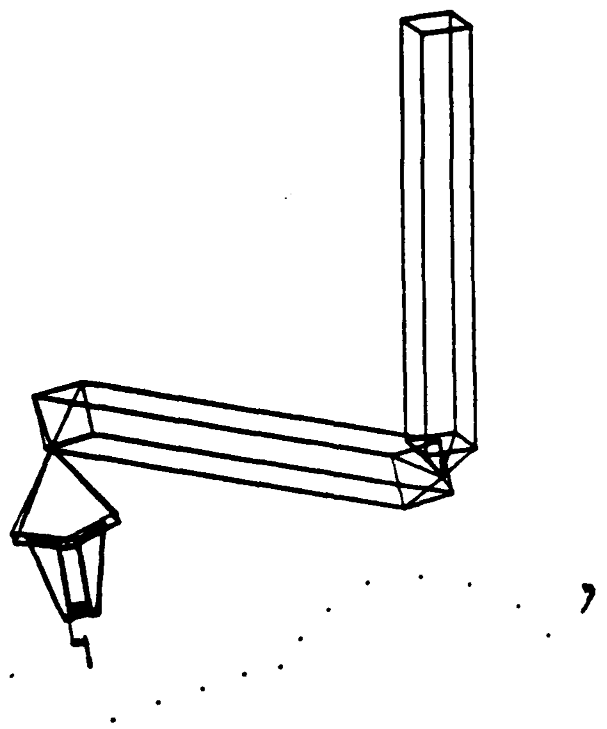


Figure 4.18 Top view of a WELD on the computer graphic display

4.10 Building Models of Manipulation Tasks Using Motions and Models of the Environment

The models of flat and curved surfaces and welds can be used together with symbolic movement commands to define a variety of tasks. Since the system has no force or other type of impedance control, the system is best suited to tasks that have simple or no kinematic constraints. This includes most of the inspection tasks that would be appropriate for an underwater teleoperator.

4.10.1 Scanning a flat surface

A general scanning action would be useful for performing cleaning operations (water jetting, brushing, etc.) on a flat surface.

The SCAN program would first move the arm to a prescribed distance in the "BACK" direction from the coordinate system defined for the plane. This insures that the hand has the proper orientation and is the prescribed distance from the plane, as shown in figure 4.20. Then the SCAN program would execute a series of relative translations to accomplish scanning action.

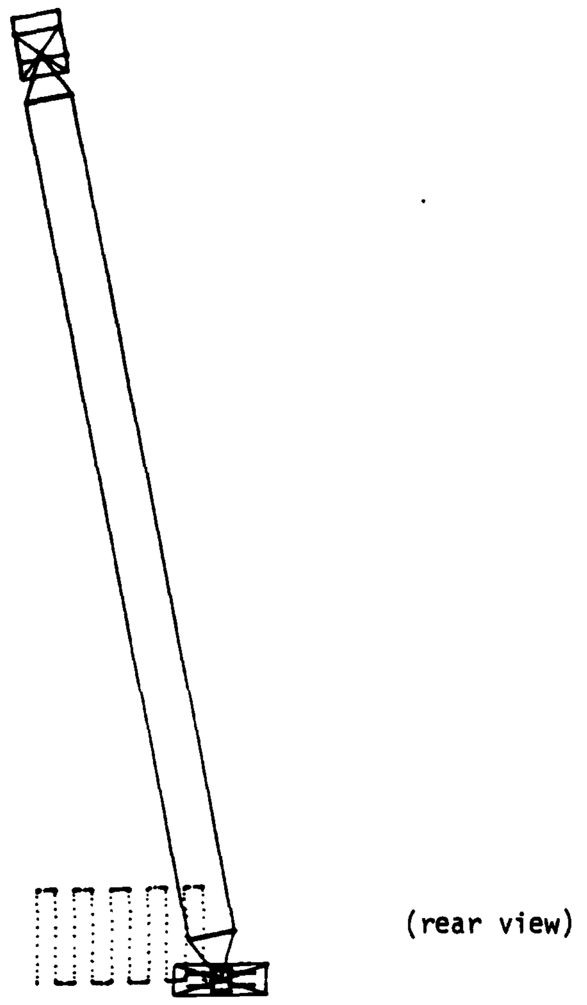


Figure 4.19 Execution of the SCAN command. The dotted line shows the path followed by the hand while executing the SCAN command.

Structure diagram:

```
            move hand to a position
            2 cm back from defined
            location of plane
SCAN
                        move hand 10 cm up
            do 5        move hand 2 cm right
            times       move hand 10 cm down
                        move hand 2 cm right
```

Source code:

```
DEFINE: SCAN    PARAM SET    2 CM BACK

                5 TIMES DO
                           10 CM UP
                            2 CM RIGHT
                           10 CM DOWN
                            2 CM RIGHT
                       LOOP ;
```

Usage:
For a PLANE named BULKHEAD:

```
                      BULKHEAD SCAN
```

4.10.3 Inspecting a weld

A task for use with WELDs is INSPECT. INSPECT is used to clean a weld with a jet or brush or to photograph a weld. Jetting or photographing requires that the arm move through a trajectory that points the hand at the weld, but stands off some distance from the weld. The distance must be changed for jetting or photographing.

This task moves the arm through the first frame defined for the WELD, then moves the arm in a trajectory defined to lie a predefined distance BACK from each of the coordinate frames defined for the WELD. The arm then departs through the last frame defined for the WELD (the departure frame).

This procedure uses both analogic and symbolic components. A weld is easy to define analogically, while a trajectory that keeps the hand a prescribed distance from the weld and still pointing directly at the weld would be difficult to define analogically. Describing a complicated weld symbolically would be very tedious. This problem is solved by defining the weld analogically, then symbolically defining a trajectory some distance BACK from the coordinate frames that make up the weld.

4.11 Monitoring Commands

Monitoring commands are used by the operator to learn about the current state of the manipulator, or to dynamically observe the arm carry out instructions. Two types of monitoring commands have been implemented. One group of commands, symbolic monitoring commands, give information to the operator in numerical form using the standard terminal. A second group of commands, graphical monitoring commands, allow the operator to manipulate the dynamic graphic display. Both viewpoint and the type of information displayed may be controlled.

4.11.1 Symbolic monitoring commands

Symbolic monitoring commands allow the operator to display the current position of the arm at the terminal. This information may be displayed in either cartesian or joint space format. The command

```
?JOINTS
```

displays the current values of the manipulator joint angles. The command

```
?WHERE
```

shows the current location and orientation of the hand frame, defined relative to the base frame. The translation is given in centimeters. The orientation is displayed as an ordered set of rotations in the sequence yaw, pitch, roll.

4.11.2 Graphical monitoring commands

Graphical monitoring commands allow the operator to change the viewpoint for the dyanamic graphic display, and to cause different types of information to be shown on the graphic display.

The viewpoint is changed by using the touch-sensitive screen of a raster graphics terminal (figure 4.3). By touching this screen at the labeled locations, the operator can cause the dynamic image of the arm to be rotated, translated, and zoomed. The column of buttons on the left of the screen correspond to fixed combinations of rotation, translation, and zoom, such as side, top, front, and standard views. The column of buttons on the right correspond to rates of translation, rotation, and zoom.

The display may also be used to show coordinate frames that have been previously defined and to show trajectories for real or simulated arm movements. As stated earlier, the analogically defined entities POSITION, PATH, REL_FRAME, and RELPATH may be displayed using the SHOW command. After the operator issues the TRACK command, the path followed by the arm will be displayed as a sequence of dots. The TRACK command works whether the system is under manual or computer control or in simulation mode.

4.12 Intervention Commands

As the human operator monitors the progress of the remote manipulator, he may decide that the system is not performing satisfactorily. The STOP command halts all computer controlled operations. The BACKUP command causes the arm to move to the previously defined frame. The BACKUP command may be used repeatedly until the beginning of a computer controlled sequence has been reached. This feature will be useful to extract the arm from a difficult position. The operator may also place the arm in any manual control mode at any time.

Chapter 5

DESIGNING SUPERVISORY TELEMANIPULATION EXPERIMENTS

5.1 Introduction

A supervisory control system cannot be designed without including the people that will operate the system. Man-machine system performance in the context of this thesis generally cannot be predicted from any available theory because of the complex interaction between human and engineering factors. In such cases, performance must be evaluated experimentally. Performing meaningful experiments involving complex man-machine systems presents major problems to the experimenter. A major dilemma concerns the ideal of performing tightly controlled experiments, versus the reality of testing an actual system without the ability to test all independent factors systematically.

5.2 Different Types of Experiments

Different researchers with different goals and responsibilities have applied different types of experimental techniques to telemanipulation problems.

Strictly controlled experiments allow precise analysis of quantitative results, but in controlling all independent psychological variables and repeating a task several times, important elements of the real world situation may be lost. However, in such an experiment it is often possible to extrapolate to a more realistic situation. This extrapolation is possible because controlled experiments can lead to an understanding of how the independent factors interact with each other. These techniques have been most successfully applied to skill-based tasks, where the task and its goals can be rigidly defined [66],[67]. Recently, these techniques have been applied to determining how people should program computers [68], [69].

Engineers are often interested in evaluating the change in performance which results from an engineering design decision, for example:

> Slow-scan versus continuous video feedback for control of a remote manipulator.
>
> Supervisory manipulator control systems with different levels of computer control capabilities.

In such cases, the design change will influence a large number of independent psychological factors. To test all design choices experimentally is not practical. Nevertheless, these are the types of design questions system designers face.

One solution to this problem is to test "realistic" combinations if all independent variables cannot be controlled and varied one at a time. However, if all independent factors have not been varied independently, the results cannot be so easily extrapolated to other situations. If a difference in performance is observed, one cannot be sure whether the difference was caused by the influence of the main effects or whether the difference was caused by the interaction of several design features. Most laboratory supervisory control experiments fall in this class [10], [55]. Nevertheless, these experiments produce very useful results within the context of the type of system being designed.

Sometimes the goals are even more basic. The experimenter may be asking "is it possible to perform this task remotely at all?" or "is this system practical?". Actual underwater tests are usually this type of experiment. Such experiments have been performed at NOSC on the Work Systems Package [70] and at the Norwegian Underwater Technology Center (NUTEC) [4]. Rather than exploring the differences between factors of psychological or engineering interest, these experiments investigate the performance of a single hardware and software configuration under real-world constraints. Performance data, such as completion time, is secondary to whether the system works at all. The most important results of these types of experiments are recommendations for improved hardware and software.

5.3 A Varied Approach

A descriptive model of supervisory control was presented in Chapter 1. This model was used to obtain a set of design goals for the supervisory telemanipulation system. Likewise, this model can be used to generate some experimental goals.

As stated in the descriptive model, in supervisory control the human operator is able to off-load skill-based and rule-based behavior to the automatic system. The supervisory function of teaching was the mechanism through which the human operator transfered skill and rule-based portions of the manipulation task to the system designed in this thesis. However, sensing and computational limitations required that some portion of skill-based behavior remain with the human operator. In particular, the human operator was sometimes required to point out a series of coordinate frames analogically, either to define a trajectory for the arm (i.e. the PATH command) or to define some feature of the environment (such as the WELD command).

After a WELD or PATH has been defined analogically, the operator may teach the computer system to execute a variety of rule and skill-based activity based on that information. The arm may be commanded to execute a trajectory along a PATH, in either forward or reverse directions, with several options for path interpolation between the defined frames. A WELD may be inspected in several ways.

While some aspects of the performance of the trajectory control system can be predicted from engineering theory and experiment, overall system performance will depend on the accuracy of the analogic information furnished by the human operator. Fortunately, this type of skill-based behavior of the human operator can be approached through controlled experimentation. If the ability of the human operator to point out coordinate frames under varying conditions can be understood, and the performance of the control system can be understood, then overall system performance can be predicted for situations for which the entire system has not been specifically tested. Controlled experiments which examine the analogic definition of coordinate frames under varying viewing conditions were run.

A second type of experiment was also performed. In this experiment, system performance was examined for different supervisory control modes. As pointed out earlier, "supervisory control mode" is not a single independent psychological factor. Each mode may differ from the others in terms of several sensory, cognitive, and motor aspects. Thus, this experiment was controlled from the point of view of engineering considerations rather than purely psychological factors.

For the second experiment, a task was designed which represents a realistic element of underwater inspection, but for which performance could be fully defined. Simulated cleaning and inspection of a complex weld could be performed manually or under several forms of supervisory control. Two different error measures were defined which were of psychological and engineering interest.

Chapter 6

EXPERIMENTAL INVESTIGATION OF ANALOGIC DEFINITION OF COORDINATE FRAMES

6.1 Introduction

A cartesian coordinate frame can be defined analogically by pointing out the frame with the manipulator. The system presented in this thesis allows frames to be defined relative to the manipulator base (absolutely, in Brooks' nomenclature), or relative to an arbitrary reference which is also defined analogically. Indicating a coordinate frame by moving the arm to the frame to be defined will be called single movement pointing. This chapter will attempt to quantify the accuracy of single movement pointing under remote viewing.

Single movement pointing is a simple, reliable method for defining position and orientation. However, the accuracy of this method is not great, particularly with regard to orientation. The experiments described here will attempt to quantify the magnitude of these errors and determine the source of these errors.

The lack of accuracy for orientation is very important if relative translations are to be executed from an analogically defined frame, as shown in figure 6.1. The accuracy of pointing can be improved by using three pointing movements to establish the frame [56], as discussed in Appendix F. This multiple movement technique is used by this system in

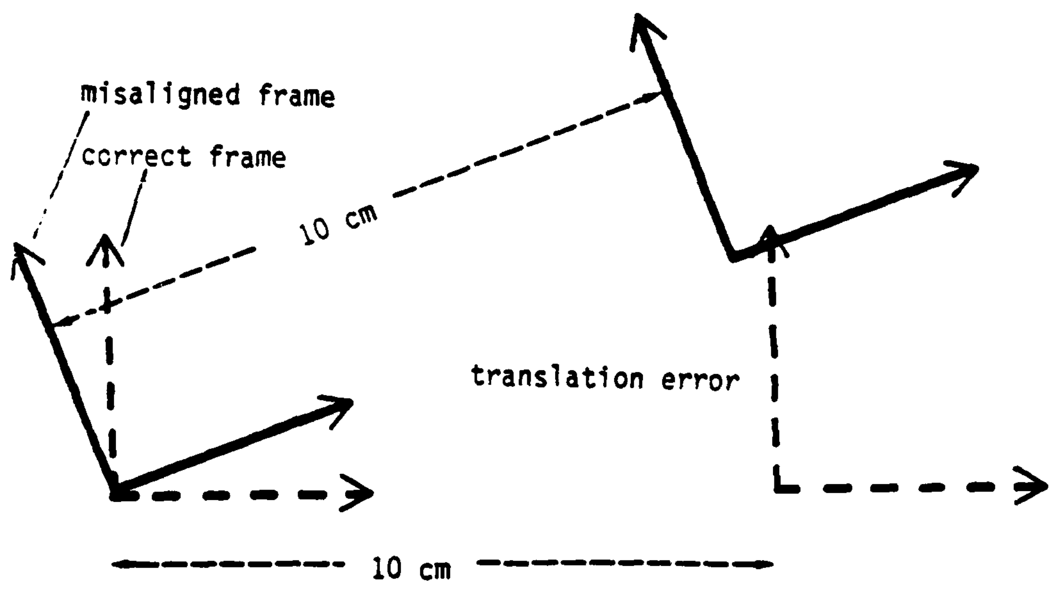


Figure 6.1 When defining a base coordinate system for relative motions, small orientation errors can produce large translation errors.

the PLANE command, described in Chapter 4.

Other sensing systems, such as laser or acoustic rangefinders [71] may also be used to gather this information. However, single movement pointing is by far the simplest. This technique requires only position feedback from each joint. Since position feedback is often included to achieve acceptable performance for manual control, a system which incorporates pointing as a teaching technique may require no additional sensors, but performance may be greatly improved. Improvements in system performance which result from including pointing into a supervisory control system will be the topic of the following chapter. Single movement pointing has been demonstrated in the water by investigators at NUTEC [4]. This chapter seeks to establish the limitations of this technique from an accuracy standpoint.

These experiments began from asking a seemingly simple question: "How accurately can someone position and orient a remote manipulator?". A careful examination of the process revealed that this question is much too poorly defined to be answered. The factors which one may expect to influence human performance in such a task may be divided into two groups, perceptual and motor. The perceptual factors include:

1. type of viewing system, including resolution, contrast, etc.

2. the lighting condition.

3. the spatial relationship between the position and orientation of the frame to be defined and the operator's viewpoint.

Motor factors include:

1. manipulator control system (master slave, RMRC, joint rate, etc.)

2. human motor ability would be important for master slave operation.

Single movement pointing is highly dependent on the viewing conditions, as it is through visual feedback that the operator aligns the manipulator hand to indicate the desired frame. While several experimental studies have looked at the influence of the viewing system on translational accuracy and on performance times for simple tasks [72] [73], no studies have been concerned with orientation accuracy.

In this chapter, quantitative results will be presented which describe the orientation accuracy of single movement pointing. The goal of this exercise is to allow this technique to be matched to tasks with compatible accuracy specifications.

6.2 Direct Viewing Experiment

6.2.1 Goals of the experiment

This experiment was conducted in the Man-Machine Systems Lab at MIT using the Argonne E-2 master slave manipulator. No television system was used. The operator could view the remote arm directly.

One goal of the experiment was to provide quantitative values of the mean and variance of errors in this task. Another goal was to determine how some of the factors listed earlier influence the accuracy of this process for a simple viewing condition.

6.2.2 Experimental Design

The experimental task was to orient the manipulator hand normal to a plane at a specified location on the plane. Three different planes were used. Each was inclined by a different angle, as shown in figure 6.2. Each plane could be in any of three different positions, as shown in figure 6.3. Each position was equidistant from the base of the slave manipulator. On each plane, five locations were marked. Inclination of the plane, position of the plane, and locations on the plane were taken to be the independent variables.

A 3x3x5 full factorial within-subjects design was used. For this type of design, each subject performs the task for each combination of the

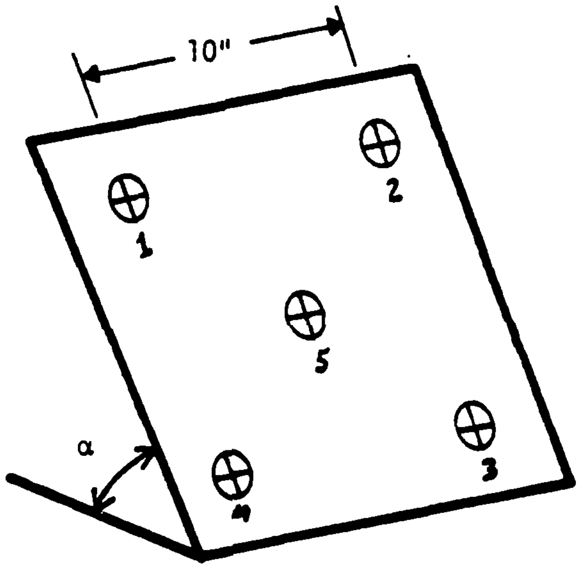


Figure 6.2
Three different planes were used in the experiment. Each was inclined at a different angle α to the horizontal plane. Values of α used were 30, 45, and 60 degrees. On each plane, five locations were marked.

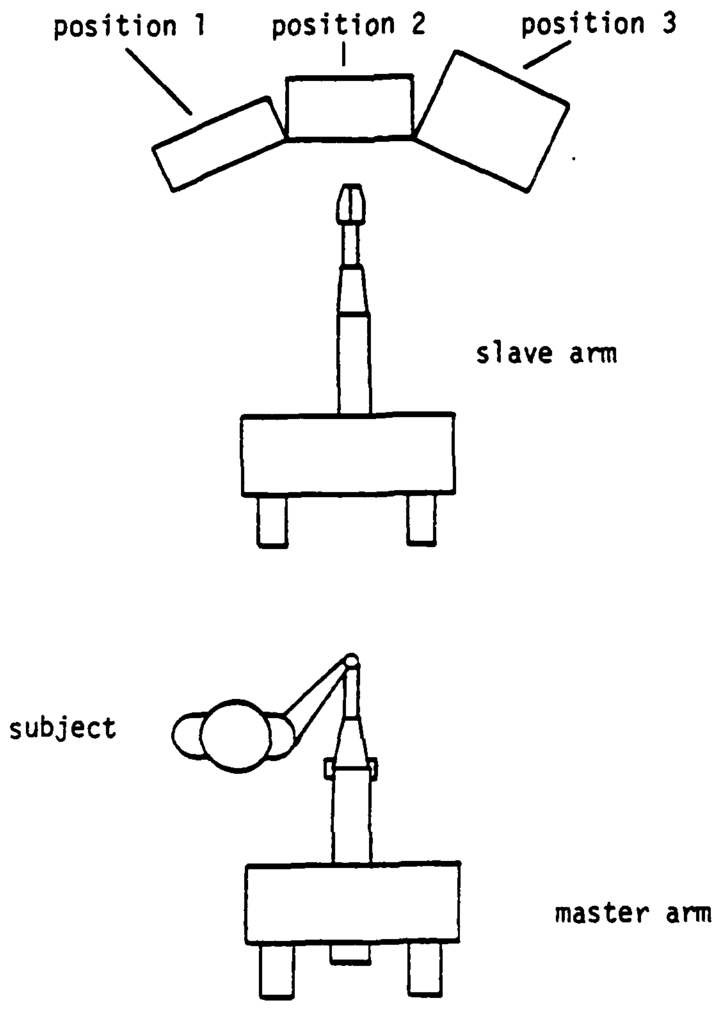


Figure 6.3 The direct viewing experiment: each plane could be placed in any of three positions. Each position was equidistant from the manipulator base. The center position was directly in front of the

independent variables. The within-subjects design was chosen because it allows effects to be observed for a small number of subjects despite interaction between subjects and main effects [67].

Three subjects were tested, all right handed male engineering students with normal or corrected vision. Each subject performed three repetitions at each combination of inclination, position, and location, so each subject performed a total of 135 trials in a randomized order. The trials for each subject were broken up into three blocks, with each block consisting of a fixed combination of inclinations and positions. The order of these blocks was counterbalanced across subjects.

Three dependent variables were considered: elapsed time, and two dimensions of orientation error.

All analysis of orientation was done by first converting the recorded joint angle values to cartesian transformations. A cartesian representation is superior to a joint space representation, as biases in the measurements due to the kinematics of the arm are removed.

The cartesian frames indicated by the subjects were compared to the measured values for that combination of the independent variables. The comparison was done by computing the rotation vector which would transform the frame indicated by the subject into the frame computed for the appropriate reference values.

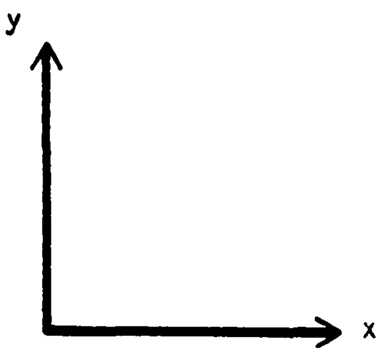
In particular, two components of angular error were analyzed. These two components are illustrated in figure 6.4. The components describe the projection of a unit vector attached to the manipulator hand into the actual plane to be defined. Together, these two components give the magnitude and direction of misalignment. They will be called x error and y error. The magnitude of error, radial error, was also computed.

The third component of orientation error, rotation about the hand was not used. For welding or inspection, this angle is probably not important, and this angle is not defined by the requirement that the hand be located normal to the plane.

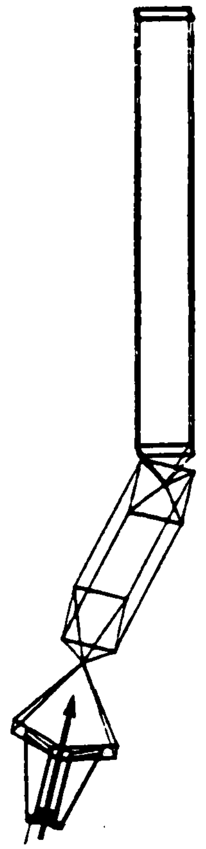
6.2.3 Procedure

Each subject was given the same written description of the task and how his performance would be judged. The instructions emphasized that accuracy was the prime performance measure, although performance time would also be recorded.

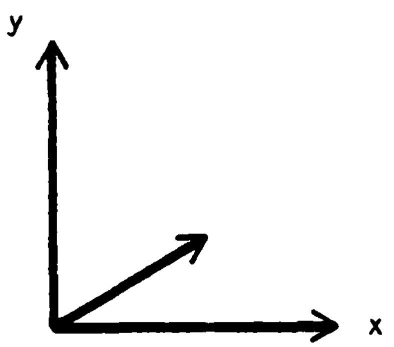
Each trial began with the master arm locked in computer control in the same position. The experimenter then told the subject at which plane and location on that plane the subject should position and orient the arm. The manipulator was then placed in manual control and the timer was started. The subject indicated when he had positioned and oriented the arm to his satisfaction by depressing a hand-held pushbutton switch. The timer was then stopped, and the current values of the manipulator's



Perfect alignment, the projection of the hand vector falls at the origin of the x y coordinate system to be defined



The hand vector is defined to be a unit vector pointing along the hand



Imperfect alignment, the hand vector has finite x and y components in the plane to be defined

Figure 6.4 Orientation error criteria

joint angles, the elapsed time, and the commanded position and location were recorded.

The correct orientation of each location on each plane was determined using the manipulator arm, but with a method much more accurate than visually orienting the hand. A special orientation tool was placed in the gripper. This tool consisted of a handle which mated with the gripper, and a flat surface which was oriented normal to the gripper. The values of the manipulator joint angles were recorded with the orientation tool placed at each location of each plane.

6.2.4 Results

The data was analyzed in several ways. Independent analyses of variance were performed for x and y error, and for elapsed time. Two dimensional plots showing mean and principal axes of variance were generated using a raster graphic display, as described below.

For the time measure, there was only one significant main effect. The analysis of variance showed that the influence of location in the plane was significant ($F(4,8)=5.4$, $p < .025$). A Newman-Keuls post-hoc analysis [67] showed that location 5, at the center of each plane, showed significantly shorter times than locations 3 and 4 ($p < .05$). Means for each location across both positions and planes are plotted in figure 6.5.

For x error and y error, a method was devised for displaying both the mean and variance in a meaningful way. Examination of the data shows

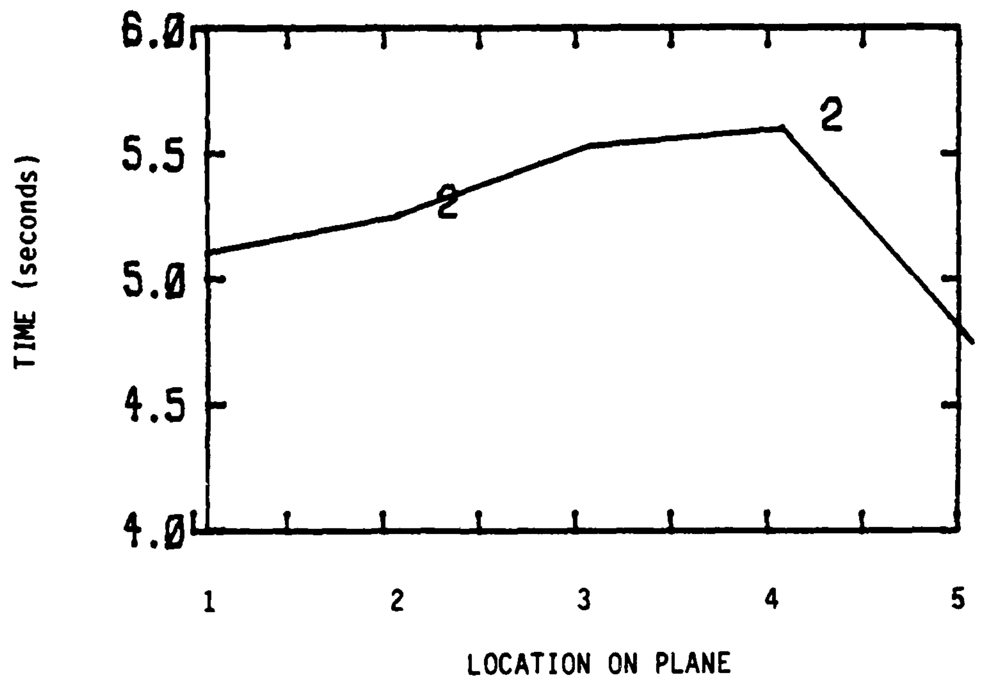


Figure 6.5 Elapsed time as a function of location. Location 5 showed significantly shorter time than locations 3 and 4.

that the error does not vary independently in x and y. The variance of this two dimensional error is best described by a covariance matrix:

$$\underline{\overline{C}} = \begin{vmatrix} \sigma_x^2 & \sigma_{xy} \\ \sigma_{yx} & \sigma_y^2 \end{vmatrix}$$

where the diagonal elements are the variances, and the off-diagnonal elements are the covariances. In general, it is possible to find a set of coordinates for which the covariances are zero [74]. The angle of the principal axes may be computed from the relationship:

$$\Theta = \frac{1}{2} \tan^{-1}\left(\frac{2\sigma_{xy}}{\sigma_x^2 - \sigma_y^2} \right)$$

The values of the variances along these axes may be called the principal variances. The principal variances are uncorrelated measures of the spread of the data. The values of the principal variances may be found by the relations:

$$\begin{vmatrix} \sigma_x^2 \\ \sigma_y^2 \end{vmatrix}' = \begin{matrix} \cos\theta & \sin\theta \\ \sin\theta & \cos\theta \end{matrix} \begin{vmatrix} \sigma_x^2 \\ \sigma_y^2 \end{vmatrix}$$

The variability of the data may be summarized by plotting an ellipse centered at the mean value of x and y error, with the major and minor axes of the ellipse equal to the square root of the principal variances (principal standard deviations). An example is shown in figure 6.6. This plot provides a descriptive "error footprint" for the data. These error plots (figure 6.7-6.9) will be useful when discussing the results of the analysis of variance.

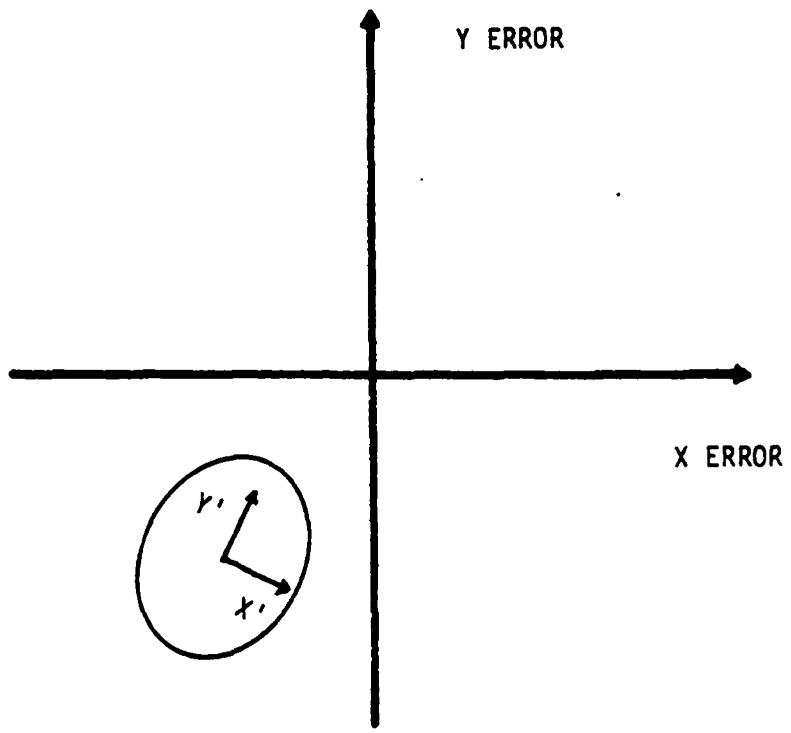


Figure 6.6 An error footprint plot. A coordinate system, x'y', centered at the mean values of x and y error, was found for which there was zero covariance in the data. The standard deviations in this new system are independent measures of the variability of the data. The variability was represented by plotting the independent standard deviations as the major and minor axes of an ellipse.

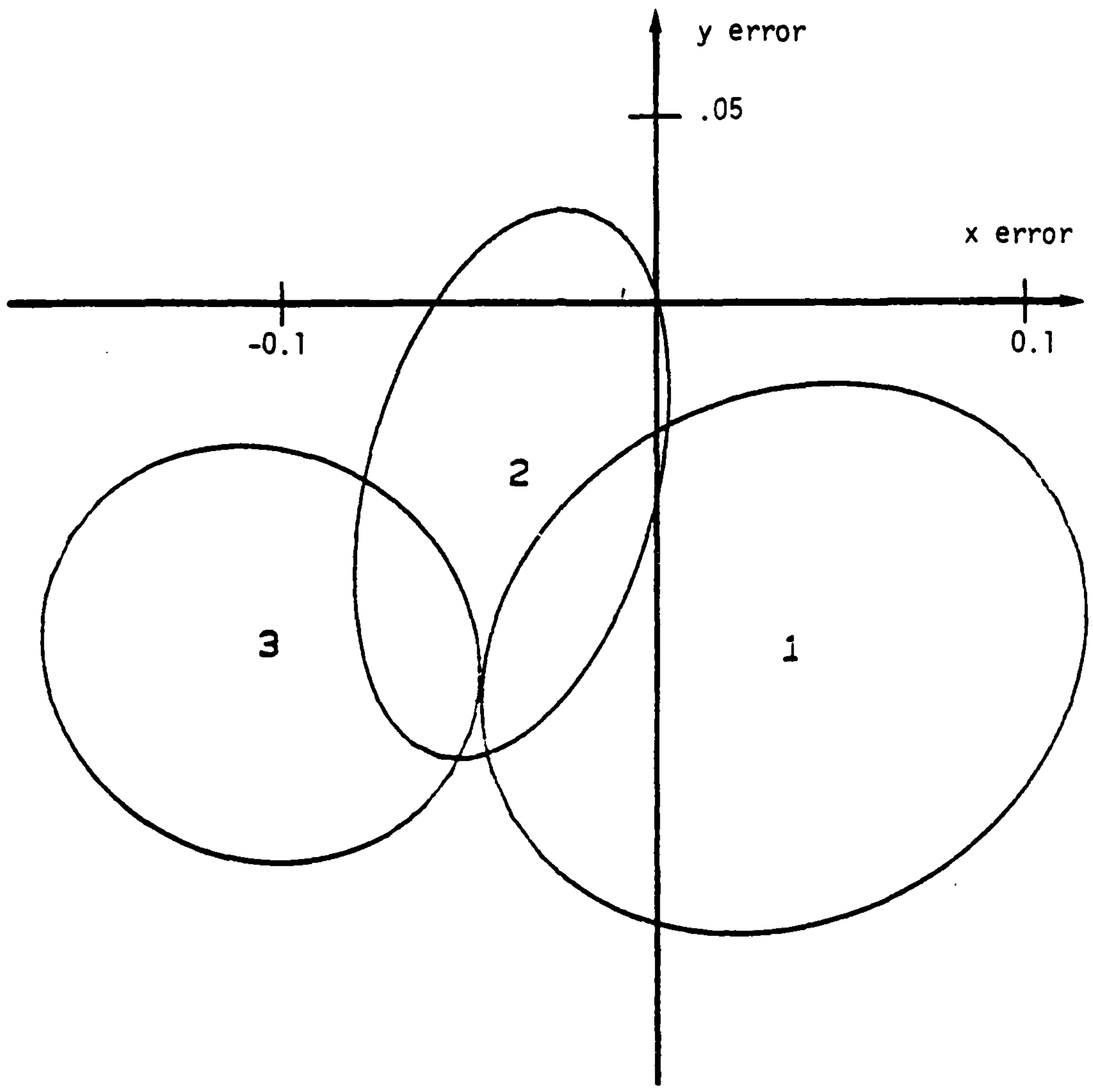


Figure 6.7 Orientation errors by position of plane. Positions 1 and 3 showed significantly higher y and radial error than position 2. For x error, all positions differed significantly.

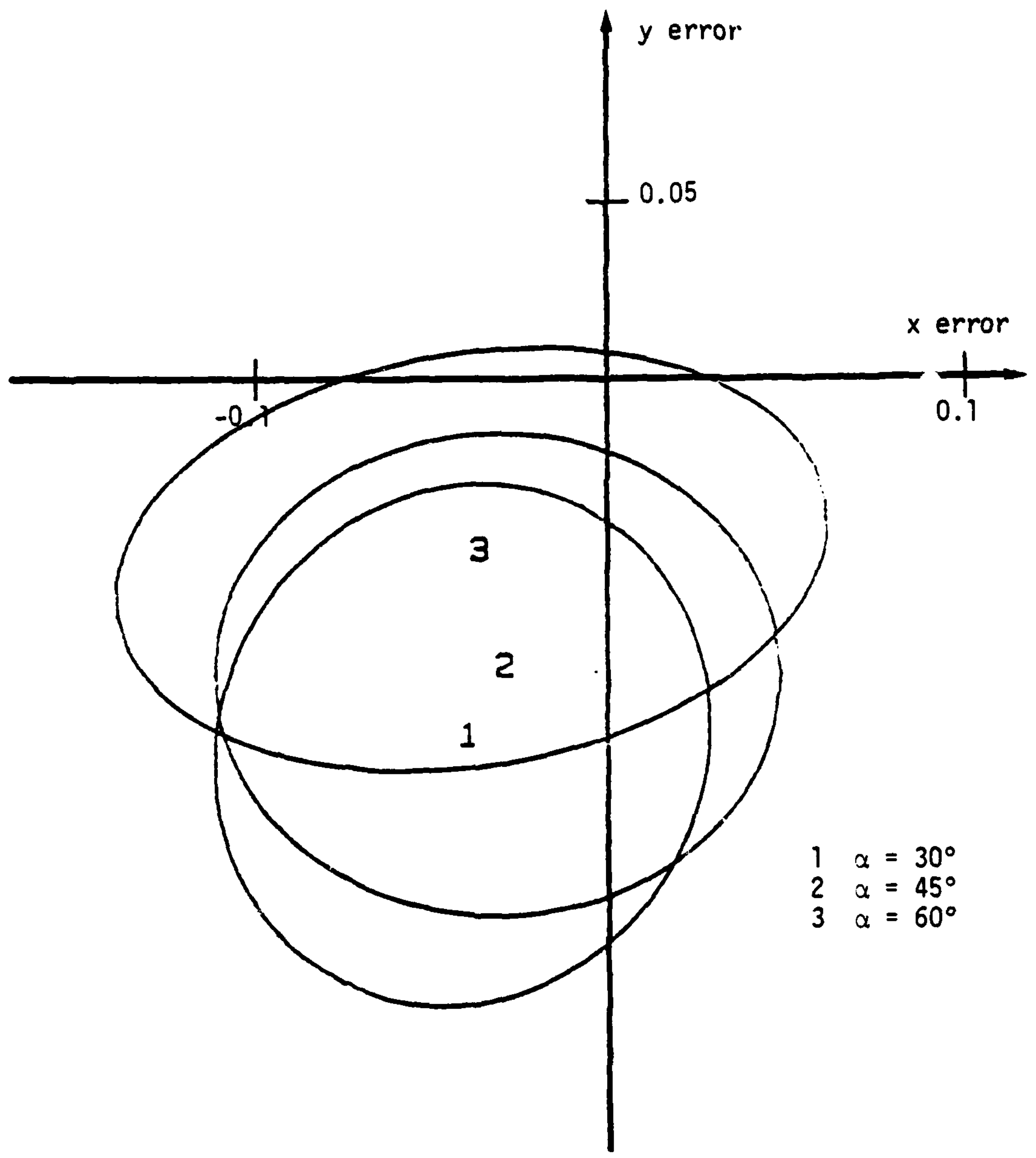


Figure 6.8 Orientation errors by inclination of the plane. The differences here were not significant, although y and radial error do correlate with inclination.

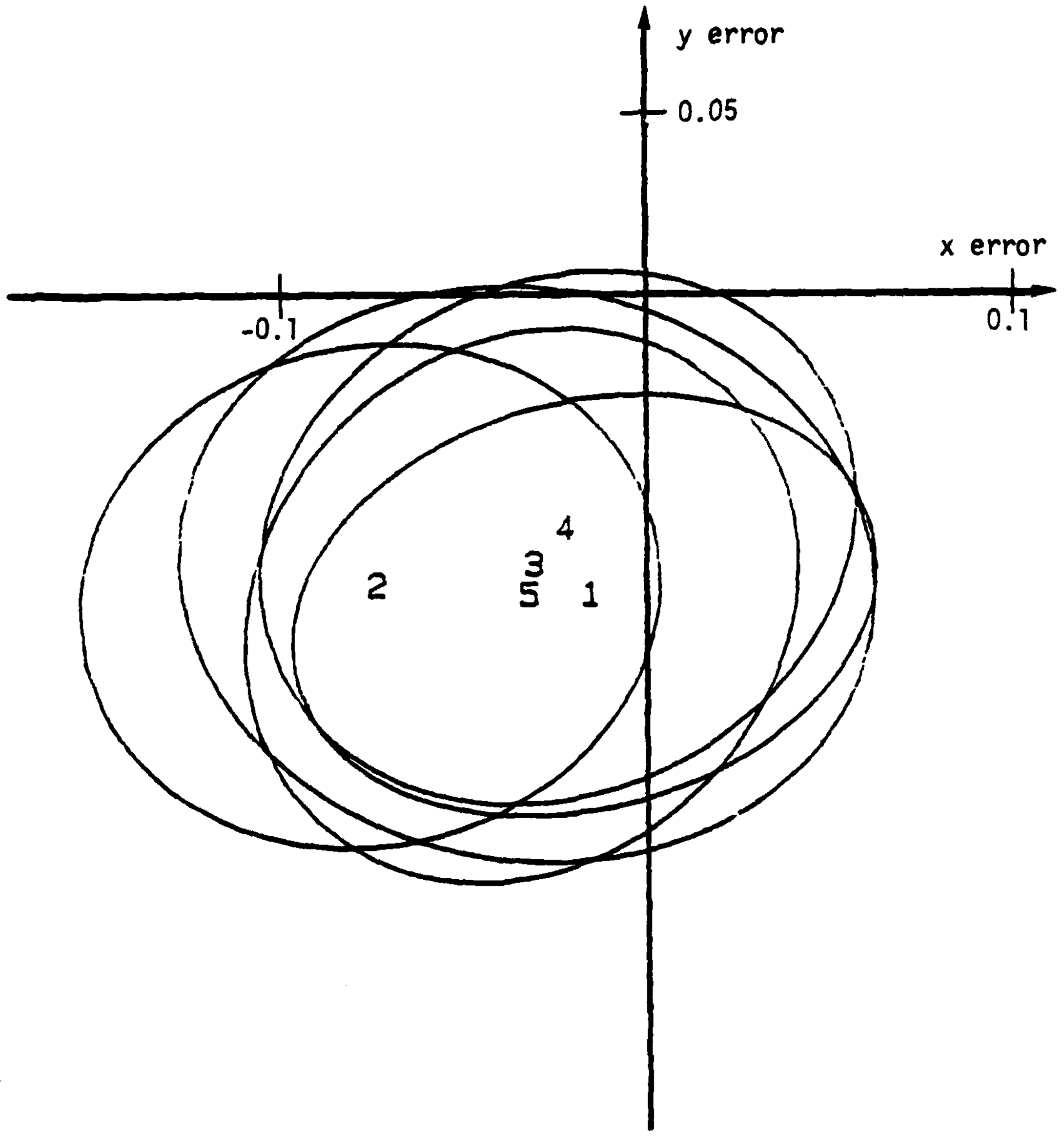


Figure 6.9 Errors by location on plane. No significant effect was seen.

Analysis of variance for x error showed significant effects for both position ($F(2,4)=19.9$, $p < .01$) and location on plane ($F(4,8)= 11.9$, $p < .01$). A Newman-Keuls [67] post-hoc test was used to test for significant differences between individual position means. This test showed that the mean x error was significantly different for each of the three positions.

The effect of position was significant for both y error ($F(2,4)=8.09$, $p < .05$) and radial error ($F(2,4)=14.3$, $p < .025$). For y and radial error, there was no significant difference between positions 1 and 3 (the left and right positions). Both positions 1 and 3 differed significantly from position 2 (the center position).

The grand mean and variability is shown in figure 6.10.

6.2.5 Discussion

Several of these statistical results have straightforward interpretations which lead to a better understanding of the sources of these errors.

The most interesting main effect is position. Interpretation of the error data for this main effect can shed light on the relative importance of perceptual and motor considerations. The left and right positions, positions 1 and 3, are similar from a perceptual point of view, as the operator was positioned directly between these two

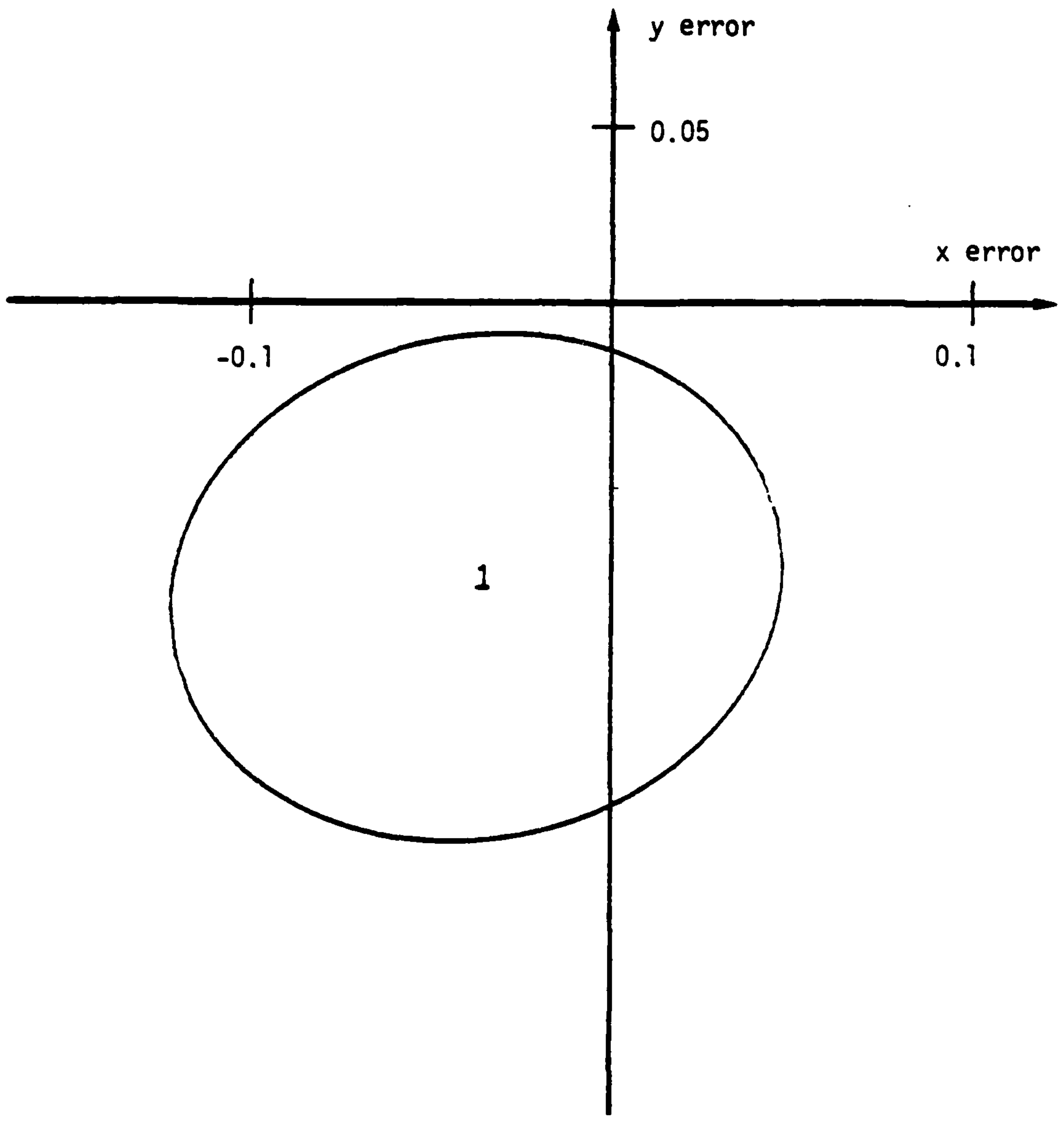


Figure 6.10 Orientation errors across all inclinations, positions, and locations.

positions. Position 2, the center position, was directly in front of the subjects, quite different perceptually from both position 1 and position 3. From a motor point of view, positions 1, 2, and 3 are all quite different, as the master slave manipulator is a right-handed device.

Figure 6.7 indicates that y error is different for both positions 1 and 3 compared to position 2, as confirmed by the statistical analysis. This is consistant with the interpretation that this dimension of error is dependent on perceptual considerations, and that no motor dependence was seen.

Figure 6.7 indicates that x error was different for all three positions. The analysis showed these differences to be significant. Another interesting observation is that both position 1 and position 3 differ from position 2 by the same amount. An interpretation of this data is that position 1 and 3 differ from position 2 due to perceptual effects, but all three means are shifted to the left due to common motor considerations.

6.2.6 Conclusions of direct viewing experiment

The following conclusions about orientation error performance under direct viewing can be made from this experiment:

1. Reliable estimates of the mean and variability of angular errors were obtained. This is helpful in evaluating when

single movement pointing is sufficiently accurate for a given application.

2. For master-slave control of the manipulator, the results are consistant with the idea that perceptual considerations dominate both the magnitude and direction of the errors over motor factors.

3. In general, orientation error is biased toward the viewpoint. This was clear from the significant difference of different positions for both x and y errors, and suggested by y error for different inclinations.

4. Since perceptual effects seem to dominate, the next step would be to evaluate how viewing through television influences these results.

6.3 Television Viewing Experiment

6.3.1 Goals of the experiment

The purpose of this experiment was to test human performance in single movement pointing while observing the position and orientation to be defined from different viewing angles. The previous experiment looked at the effect of position and orientation of the defined frame for a fixed viewpoint. In this experiment, one orientation was tested for different viewing angles.

6.3.2 Design

The task was the same as in the direct viewing experiment, but television viewing was used. The camera position was defined in terms of two angles of a spherical coordinate system with constant radius. Four different camera positions were used, as shown in figure 6.11. A sample of the television picture is also shown. Only one plane in a fixed position was used, as the previous experiment indicated that motor considerations are less important than perceptual factors. There were 4 locations marked on the plane. Camera position, location on the plane, and the practice effect were the independent variables. The experimental setup is summarized in figure 6.12.

A full 4x4x2 within-subjects factorial design was used. Eight subjects were tested. Each subject performed 8 blocks of 16 trials. Within each block, the subjects made 4 trials for each location on the plane in a different randomized order. After each block, the camera position was changed. The first four blocks corresponded to the first phase. In the first phase, the subject was given feedback about his performance from the graphic display after performing the trials for each block. In the second phase, the subjects performed blocks for each camera position again, but without feedback from the error display. Within each phase, the order of camera positions were counterbalanced across subjects. Each subject was a right handed engineering student with normal or corrected vision.

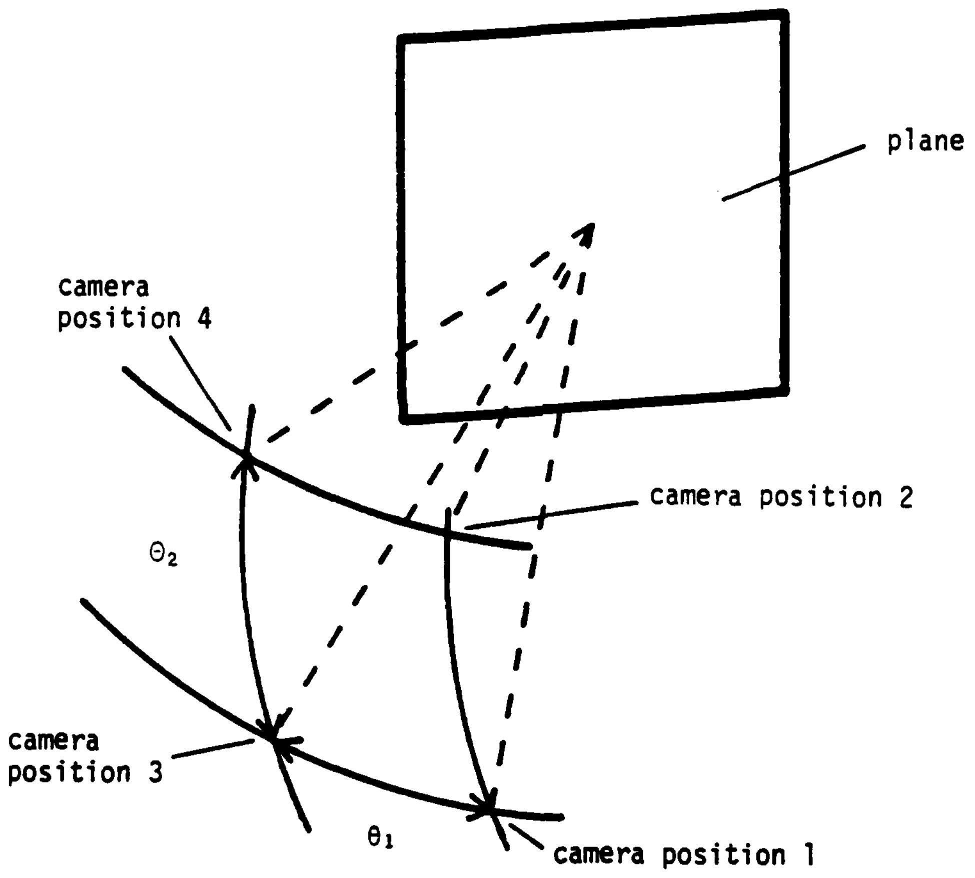


Figure 6.11 Camera positions were defined in a spherical coordinate system based at the center of the plane to be defined.

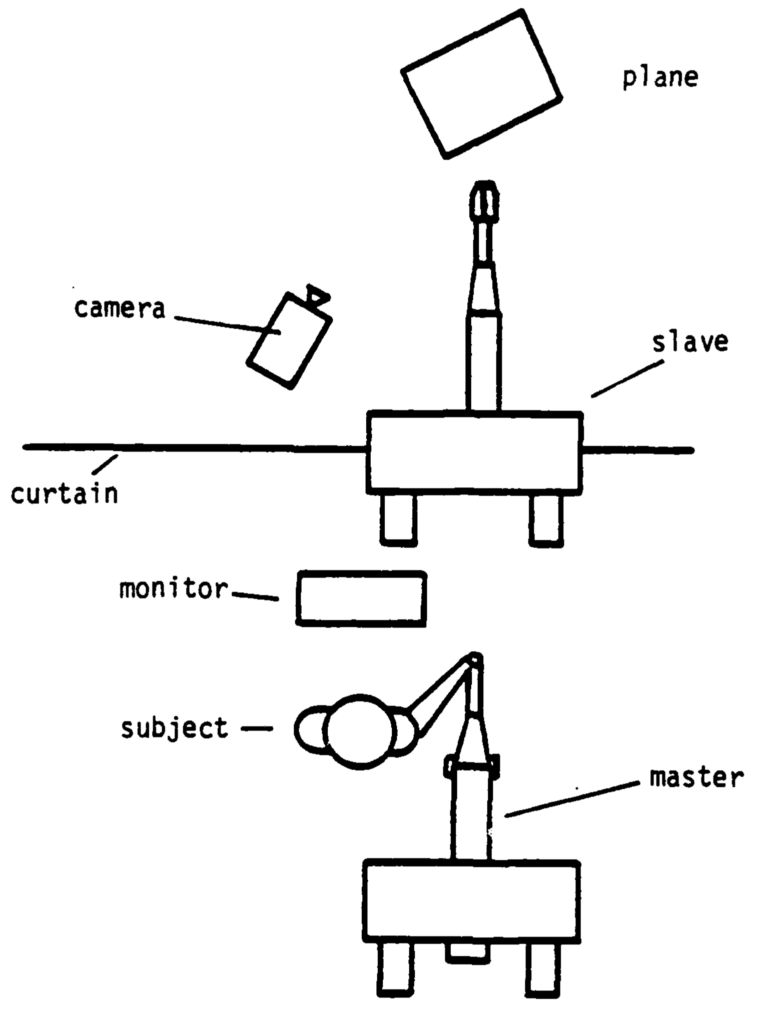


Figure 6.12 Setup for the television viewing experiment.

The dependent variables were elapsed time and orientation errors as defined in the previous experiment.

6.3.3 Procedure

Each subject was first given written instructions, emphasizing the same performance measures as in the direct viewing experiment. Subjects were also given written instructions about the meaning of the error display. Each trial proceeded in the same manner as in the earlier experiment.

6.3.4 Results

Analysis of variance was performed on both x, y, and radial error and time data. Error plots were also produced, as was done in the earlier experiment.

Figure 6.13 shows the error data for phase 1 and phase 2. A small decrease in the average error is seen, but this difference was not significant in either x, y, or radial error.

Figure 6.14 shows the error data for the different camera positions. The difference was significant for y error ($F(3,21) = 41.$, $p < 0.001$). This difference in y error is shown more clearly in figure 6.15.

Figure 6.15 shows the error plots for the different locations on the plane. This difference was significant y error ($F(3,21) = 50.$, $p <$

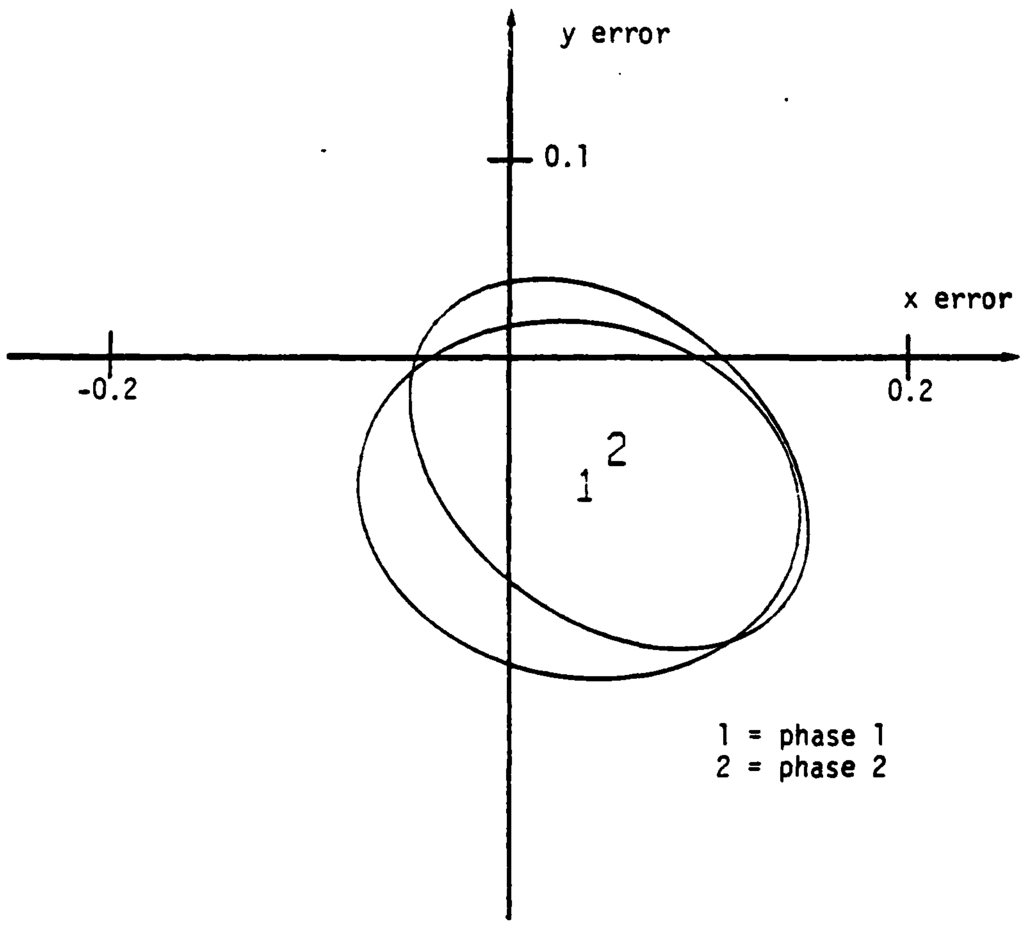


Figure 6.13 Error plots for the practice effect. A small but statistically insignificant improvment was observed in phase 2 ~~over~~ phase 1.

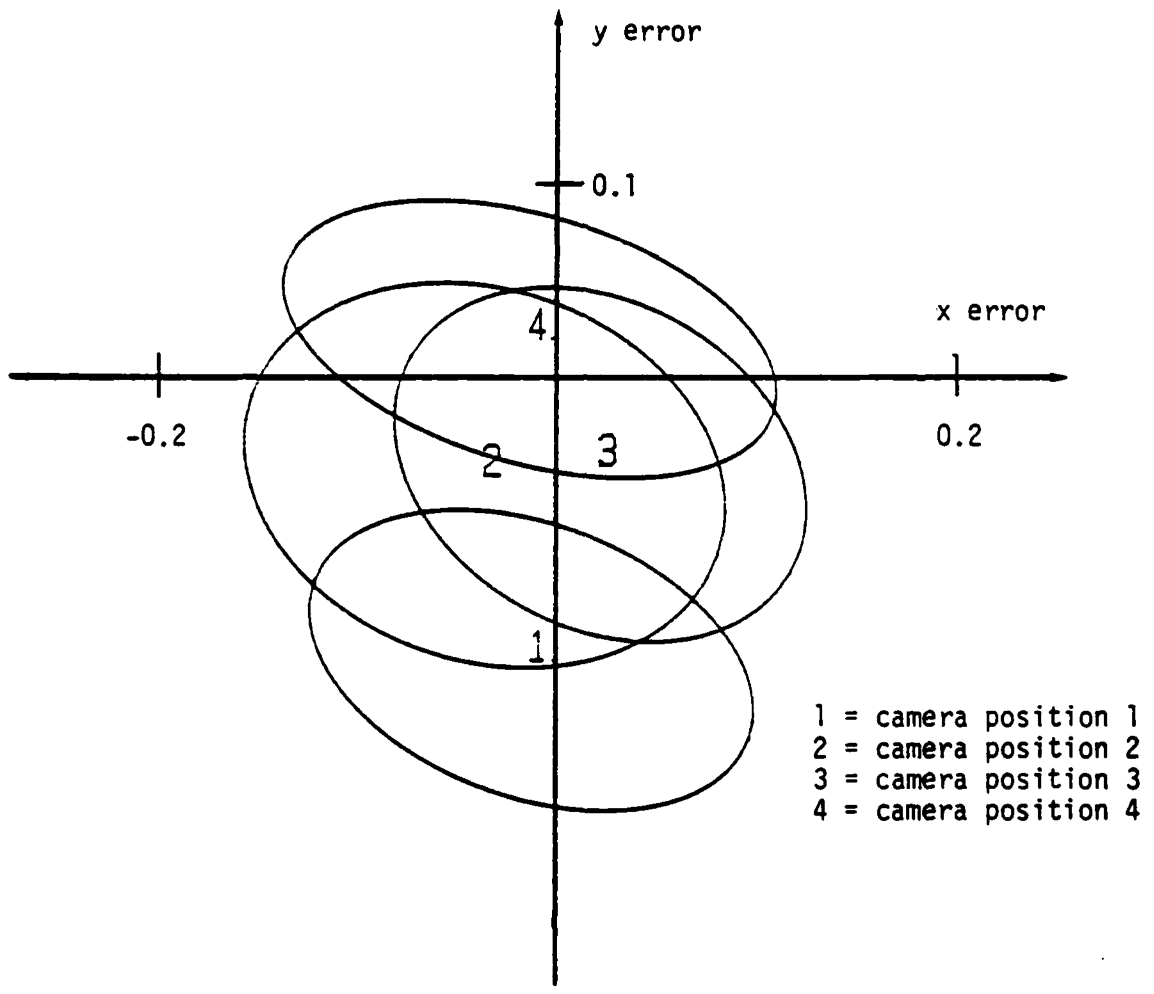
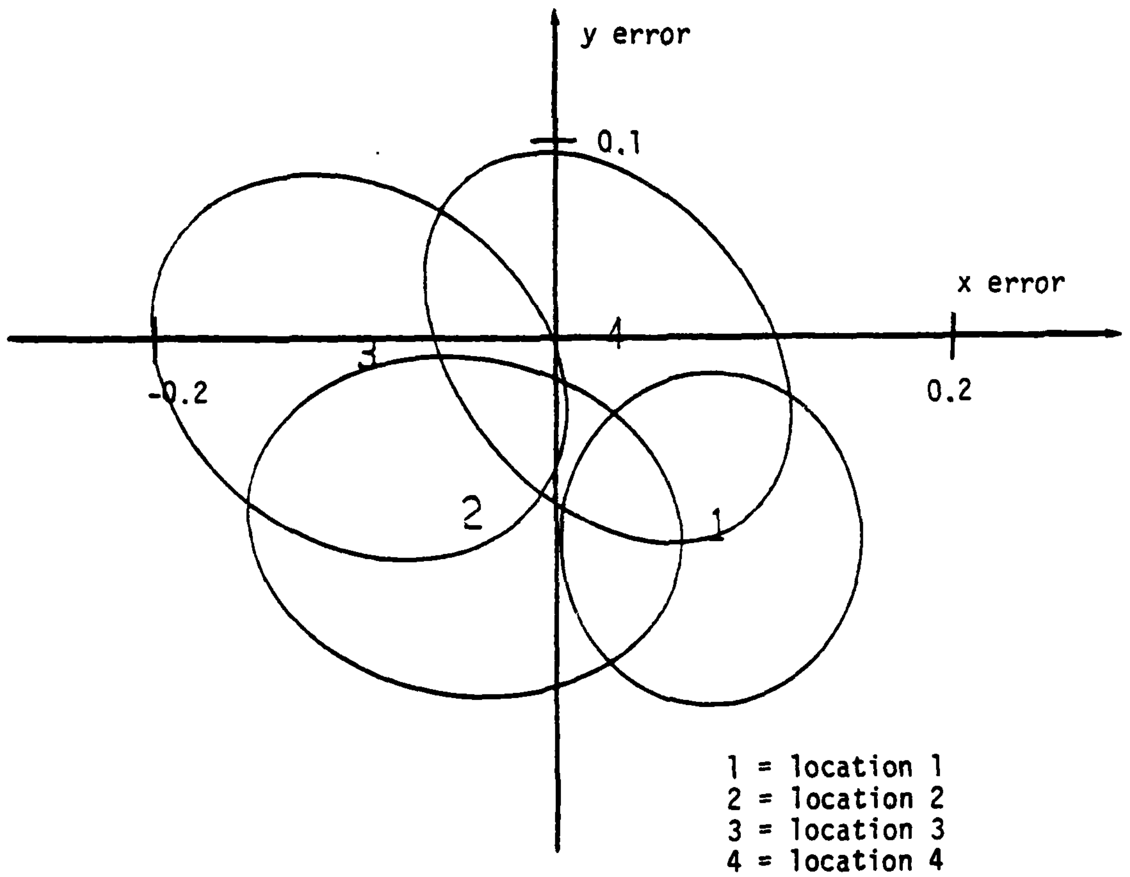


Figure 6.14 Error plots for different camera positions. The effect was significant for y error, but not for x error.



1 = location 1
2 = location 2
3 = location 3
4 = location 4

locations on the plane
were defined in figure 6.2

Figure 6.15 The effect of different locations on the plane can be seen in this plot. The effect of location on the plane on x, y, and radial error was significant.

.001). Figure 6.16 shows the means for each combination of camera position and location on the plane. This figure shows a low interaction between camera position and location on the plane. In other words, the effect of location on plane was similar for each camera position, even though the mean was different for each camera position. Although this interaction was statistically significant, it appears to be small. Figures 6.17 and 6.18 shown x and y errors for each combination of camera position and location on plane. Again, the low interaction can be seen.

Radial error, which describes the magnitude of the angular error, showed only small changes for each combination of camera position and location on plane. This plot is shown in figure 6.19.

The mean and variation averaged across all independent variables and subjects is shown with the mean and variation for similar inclination of plane, position of plane, and locations on plane in the direct viewing experiment in figure 6.20.

6.3.5 Discussion

The direct viewing experiment showed that this task is dominated by perceptual considerations. The television viewing experiment provides further clues about the source of these errors.

The significant differences in performance as a function of camera

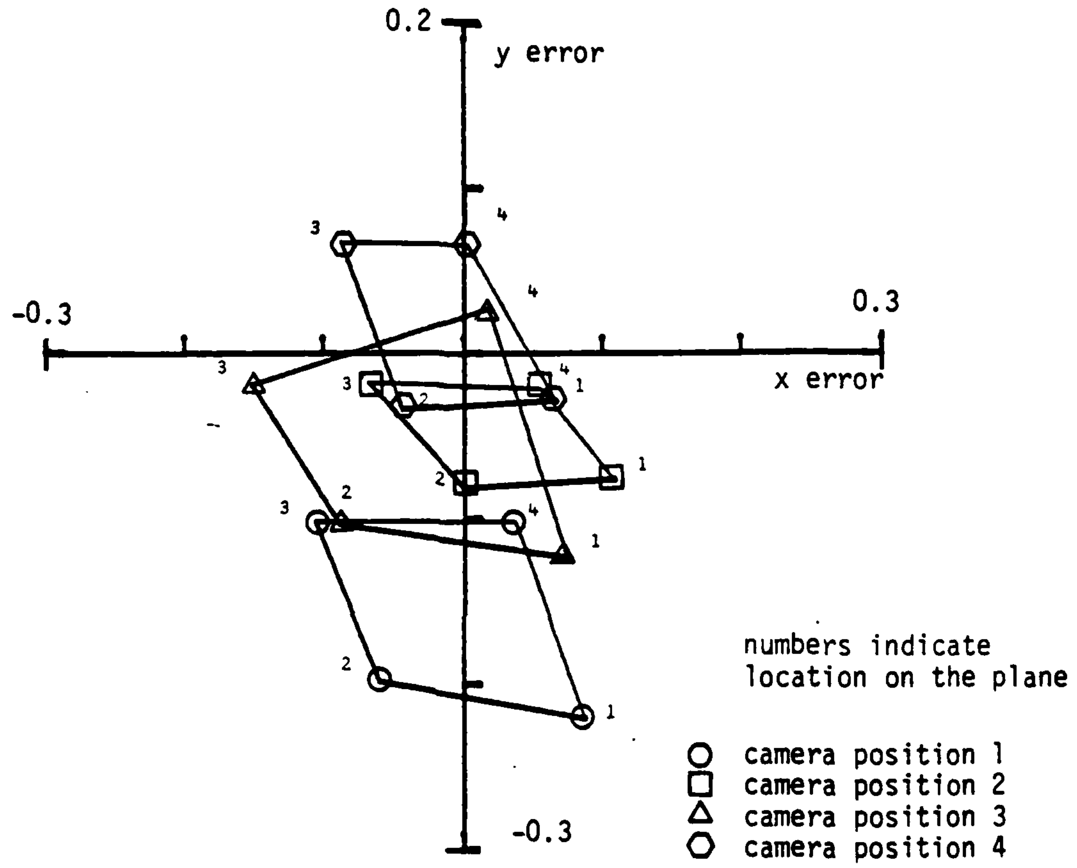


Figure 6.16 Means for different combinations of camera position and location on the plane. For each camera position, the means for each location vary in a similar way, indicating very low interaction between camera position and location on plane.

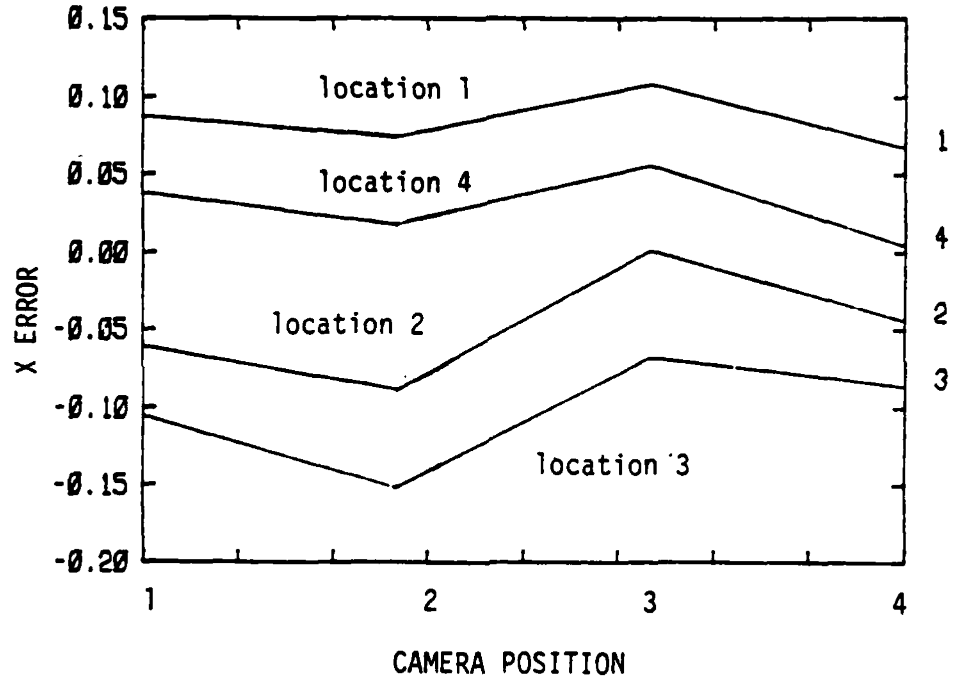


Figure 6.17 X error for different camera positions. There were no significant effects.

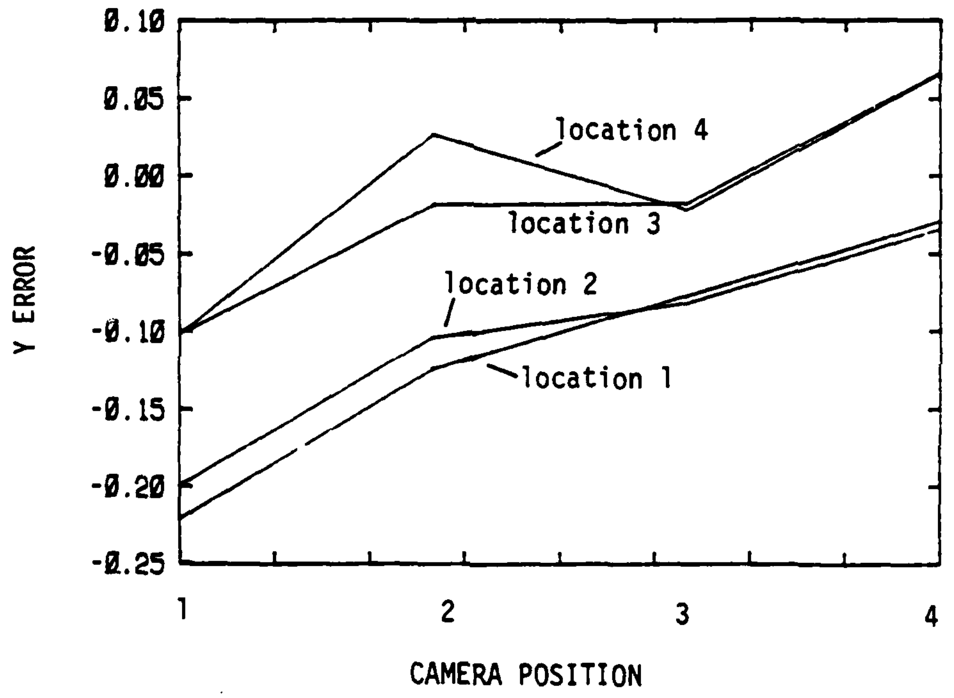


Figure 6.18 Y error for different camera positions. The effects of camera position and location on the plane were both significant. Raising the viewing angle of the camera (increasing O in figure 6.11) consistantly caused the y error to become more positive. Errors for locations 3 and 4 (on the bottom of the plane) were always more positive than errors for locations 1 and 2 (top of the plane) for the same camera position.

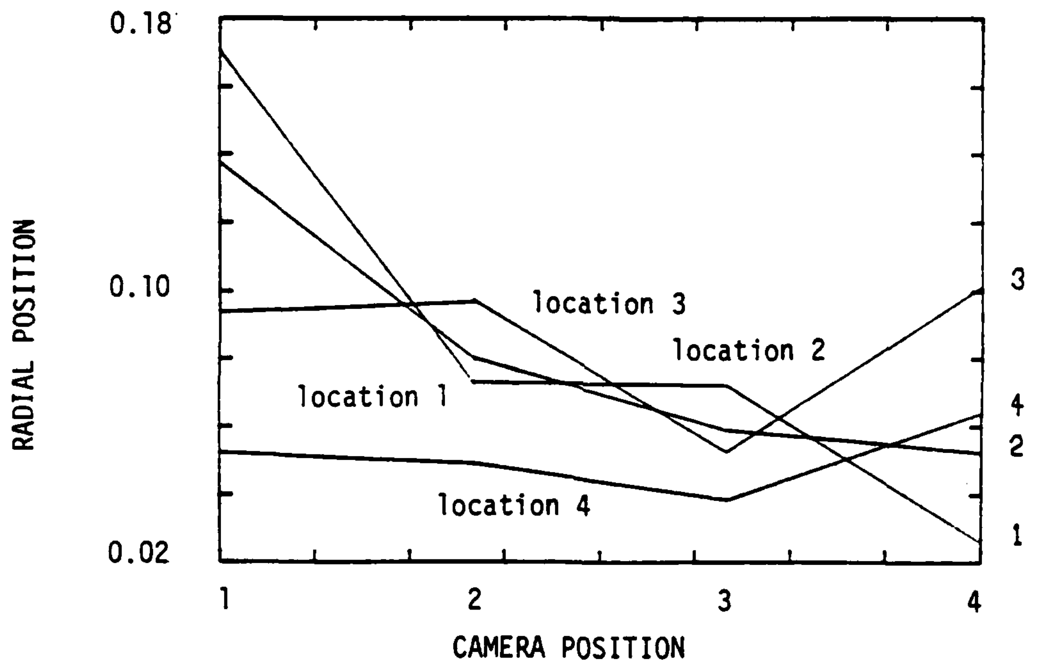


Figure 6.19 Radial error for different camera positions. Locations 1 and 2 on the plane had significantly higher radial error, otherwise there were no consistant effects.

position confirm the perceptual nature of this task. For each location (i.e. constant motor factors) y error varied significanty. The small interaction between location on plane and camera position shows that the location effect was fairly consistant for different camera positions.

The significant differences in performance for different locations on the plane can also be tied to perceptual effects and seem to support the idea that the direction of the errors are determined by the viewpoint, as seen in the direct viewing experiment. The mean error for different locations on plane (figure 6.15) are directly related to the actual positions on the plane. The errors for location 1 (upper left hand corner of the plane) are consistantly toward the lower right hand corner, and so forth.

Examination of the plot of y error (figure 6.18) shows several effects. Raising the camera always made y error more positive, while reaching for a lower location on the plane also made the y error more positive. This is the same bias toward the viewpoint which was seen in the direct viewing experiment. This corresponds to a consistant underestimation of the orientation of the plane relative to the direction of gaze, even when the direction of gaze is defined by a camera. This underestimation effect is consistant with the underestimation of radial direction constancy found by Hill [76].

Another effect that can be seen in the y error data is that for any location on the plane, error always decreased in magnitude when the camera was moved from the 22 degree position (camera positions 1 and 2) to the 45 degree position in the horizontal plane (camera positions 3 and 4 in figure 6.11). This is consistant with practical experience, including underwater tests [4].

6.4 Conclusions from direct viewing and television experiments

The following conclusions can be drawn from these exeriments:

1. While the average error was similar to the direct viewing experiment, the variation in error was larger.

2. The direction of errors were consistantly tied to perceptual issues corresponding to the spatial relationship between the plane to be defined and the direction of gaze.

3. For both direct and television viewing, subjects consistantly underestimated the relative orientation between the direction of gaze and the plane to be defined.

4. A 45 degree angle between the direction of gaze and the plane to be defined was found to have best performance.

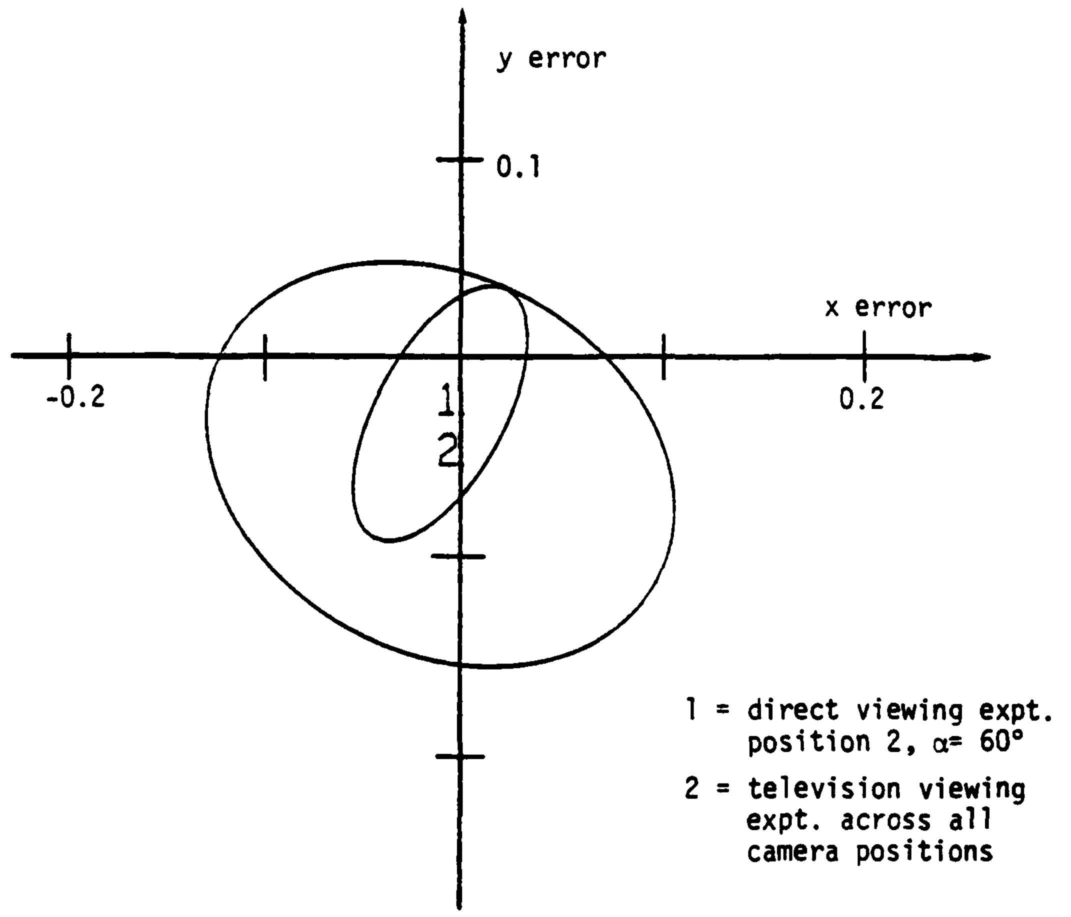


Figure 6.20 Comparison of direct viewing and television viewing experiment for similar inclination of plane, position of plane, and location on plane.

Chapter 7

REMOTE INSPECTION UNDER SUPERVISORY CONTROL: AN EXPERIMENT

7.1 Purpose of the Experiment

The purpose of this experiment was to examine how well remote inspection can be performed using three different methods for controlling the manipulator. A manual control mode (master-slave) and two supervisory configurations were compared. Performance was evaluated quantitatively in terms of time and trajectory error.

Remote inspection of a weld was chosen for the following reasons:

1. The task is realistic and is currently being examined by researchers and manufacturers of remote work systems.

2. The task is difficult to perform remotely.

3. The task includes many representative elements of supervisory control which can be measured and perhaps generalized.

Underwater inspection of welds involves two steps. First, the weld must be cleaned. This step is usually performed with a water or slurry jet, although a brush or needle gun is sometimes used. Then, some type of NDT (non-destructive testing) may be performed. All types of NDT in current use require that the weld be first cleaned down to bright metal [75]. The most common NDT method is 35 mm still photography, although acoustic and magnetic particle methods are also being developed [75].

Both steps require that the trajectory of the manipulator be controlled so that:

1. The tool (water jet, camera, etc.) points directly at the weld. As a result, the jet hits the weld, or the weld is centered in the camera's field.

2. The distance between the tool and the weld be maintained at a specified value. This requirement insures that the water jet performs properly, and that the camera remains in focus.

This type of trajectory control problem is found in a variety of inspection and maintainence tasks. A review of relevant tasks is given in Appendix H. Of the tasks described, the following represent trajectory control problems similar to the one tested in this experiment:

1. cleaning with a brush

2. grinding a weld

3. ultrasonic thickness measurements

4. high pressure jet cleaning

5. cutting concrete with a jet

In the cases of brush cleaning and grinding, compliant tools are currently being designed at NUTEC [4],[76], which would allow these tasks to be considered trajectory control problems rather than problems dominated by kinematic constraints.

This experiment examined how accurately the inspection could be performed with the three control methods. In addition to accuracy, the time required to complete the task manually versus the time to teach the system in the two supervisory modes was examined.

7.2 Experimental Design

Two supervisory control modes and one manual control mode will be compared in this experiment. These three modes are:

Mode 1. Master-slave control: this control mode is generally considered to give the best performance for any manual control mode [10].

Mode 2. Analogic teaching: The operator first teaches the trajectory to be followed as a series of discrete positions. Then, the computer can generate a smooth trajectory between those positions. This involves use of the PATH and TRAVERSE commands developed earlier.

Mode 3. Combined analogic and symbolic teaching: The operator teaches the weld (not the trajectory) as a series of discrete positions (the WELD command). Then the operator invokes a procedure (INSPECT) that generates a trajectory which is defined symbolically relative to the positions defined for the weld. The INSPECT procedure attempts to generate a trajectory that points at the weld while maintaining a specified distance between the manipulator and the weld.

These three modes represent distinct levels of computational capability on the part of the machines. In mode 1, only position servo control was required. In mode 2, the system had the ability to store positions of the arm, and servo smoothly between them. In mode 3, the system had knowledge of the forward and inverse kinematics of the arm, a capability to perform relative transformations, and the interpolation capability of mode 2.

These three modes represent the off-loading of more skill-based behavior from the human operator to the computer system. In mode 1, the subject had to determine the complete trajectory from the television picture, and then manually control the arm over that trajectory. In mode 2, the subject had to point to a number of positions and orientations along the trajectory. The system could then compute the complete trajectory by interpolating and servo the arm along the trajectory. In mode 3, the subject had to point out a series of positions and orientations along the weld. The computer system could then compute the trajectory by transforming the positions and orientations taught by the subject, interpolating, and servoing the arm.

Performance in all of these modes is dependent on both human and machine performance. Performance in mode 1 is dependent on the subjects' perceptual and motor capabilities, and is influenced by the electromechanical peculiarities of the E-2 arm. In mode 2, performance was less dependent on the dynamic properties of the arm, since the operator only needed to point out discrete positions. However, the operator needed to understand the rule-based behavior of the machine and how the system interpolated between the taught positions. In mode 3, the operator needed to understand, in at least a qualitative way, how the system transformed the taught points into the trajectory in order to choose the points properly.

A test weld was constructed, consisting of both straight and smoothly curved sections (figure 7.1). The weld bead was assumed to run along the intersection of the upper and lower halves of the test channel. The weld could be placed in any of three orientations directly in front of the slave arm. The task consisted of defining a trajectory that kept the tool 1 inch from the weld bead and pointed directly at the weld bead as defined by the following performance criteria:

1. The distance between the tip of the hand and the weld as a function of time was computed. The subject was instructed to maintain this distance at one inch. The distance was computed off-line at 10 hz. This sampling rate gave a total of about 150 data points per experimental trial. Mean and RMS for each run were computed.

2. A measure was developed for how accurately the hand was

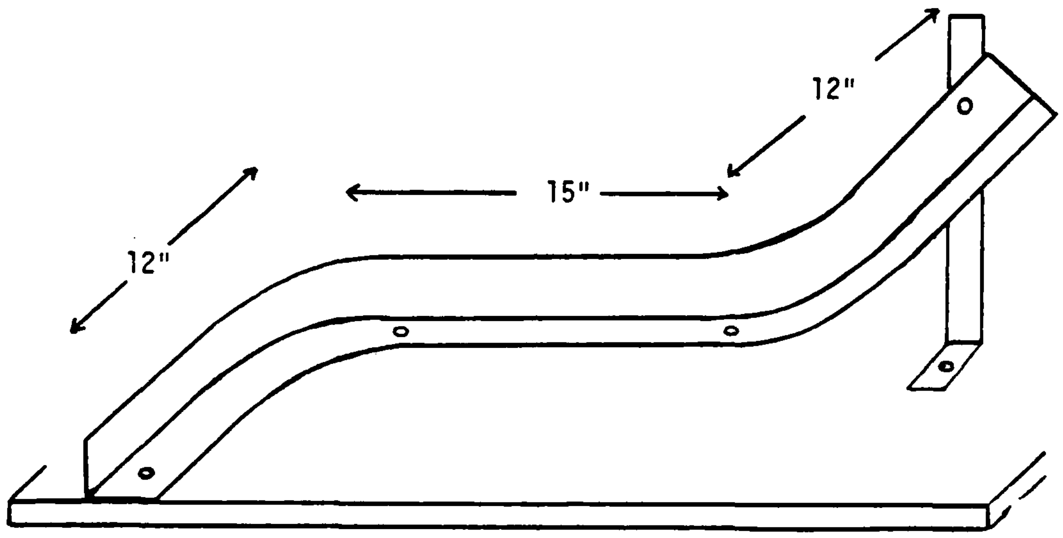


Figure 7.1 Test weld

pointed at the weld (figure 7.2). For each recorded position of the hand, the distance between the closest point on the weld and a line oriented with the hand was computed. This distance corresponds to how much the center of the water jet would miss the center of the weld. This measure was also computed at 10 hz.

Elapsed time was measured. In master-slave, the time required to complete the trajectory was recorded. In the supervisory modes, the time required to teach the required analogic information was recorded. This allowed do-it-yourself versus teach-the-system to be compared on a time basis.

Subjects viewed the test weld and the slave arm through television, with the camera position fixed. The placement of the camera, manipulator, test weld, and subject is shown in figure 7.3.

A single factor within-subjects design was used, with control mode as the independent variable. Three subjects, all right handed engineering students with normal or corrected vision were used.

Each session for each subject consisted of 3 trials in each control mode with the test weld in three different orientations for a total of 9 trials per session. The combinations of orientation and control mode were executed in a counterbalanced order. The orientation of the weld was not taken as an independent variable, but was varied to control

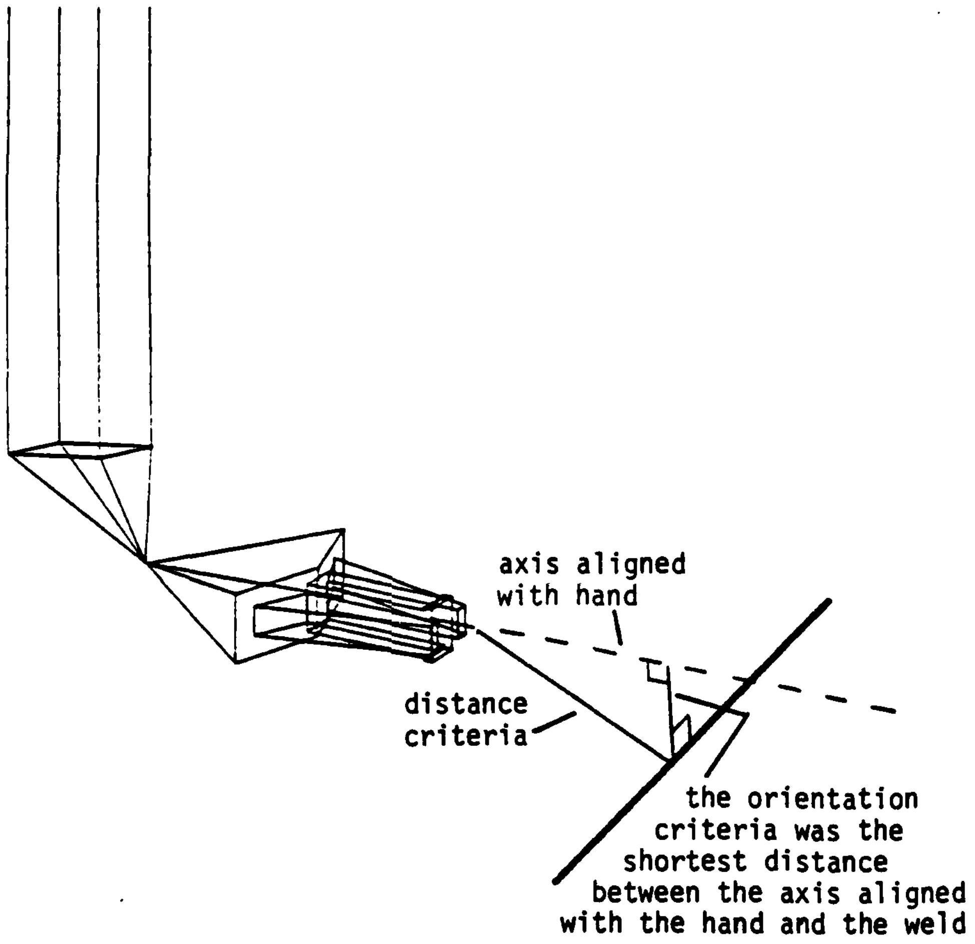

axis aligned with hand
distance criteria
the orientation criteria was the shortest distance between the axis aligned with the hand and the weld

Figure 7.2 Performance criteria. The distance criteria was the shortest distance between the tip of the tool and the weld bead. The orientation criteria was the shortest distance between the axis aligned with the hand and the weld bead.

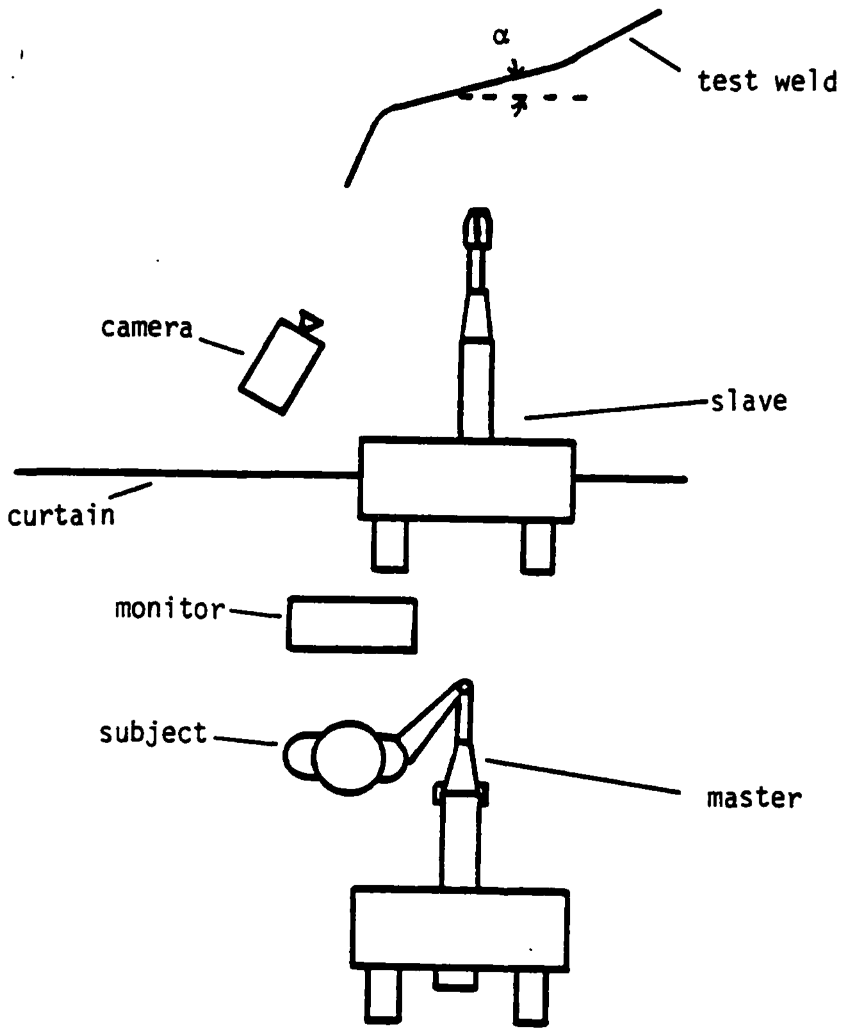


Figure 7.3 Experimental setup. Three orientations of the test weld were used ($\alpha = -15°, 0°$, and $15°$).

short term motor learning effects. Each subject performed three sessions on separate days, the first two for training and the last for data.

7.3 Procedure

Each trial began with the subject positioned directly in front of the television monitor, with the master arm control handle in his right hand. The experimenter told the subject which control mode to use and adjusted the orientation of the weld between trials.

For manual control, the subject indicated the start and finish of the trial by pushing a button with his left hand. The time to complete the trajectory was recorded. For the supervisory modes, the pushbutton was used to indicate the individual frames and the elapsed time to define the frames was recorded. After the frames were defined, the trajectory was executed and recorded.

Each time the weld was moved, a new reference trajectory was measured with the arm and stored. Reference trajectories were made by manually moving the end of the slave arm along the weld and recording the position at 5 hz. The trajectories generated by the subjects in each mode were then compared to the appropriate reference trajectory and the performance criteria computed.

The first two sessions for each subject were practice sessions. For the first practice session, subjects could view the remote area directly, for the second session and the final session, television viewing was used. After each run, subjects were shown a time plot of both performance criteria on a graphic display. An example of this display is shown in figure 7.4.

7.4 Results

First, estimates of the noise in the measurements of the performance criteria were obtained. The noise estimate was made by making two reference trajectory files of the weld in a fixed orientation. The two performance measures were then computed for these files, treating one file as the reference file and one as data from an experimental run. Since the actual trajectory was the same for both of these files, the resulting performance measures resulted from measurement noise. These tests gave an RMS error of 0.05 in the orientation measure, and 0.07 in the distance measure. The performance scores computed for runs in all modes were all substantially higher than these measurement noise values.

For each experimental run, the distance and orientation criteria were computed as a function of time and displayed on a graphics screen. Mean and RMS values for both criteria for each run were computed.

No significant effects were seen for mean errors, either for the distance or orientation measure. The subject means for this data is

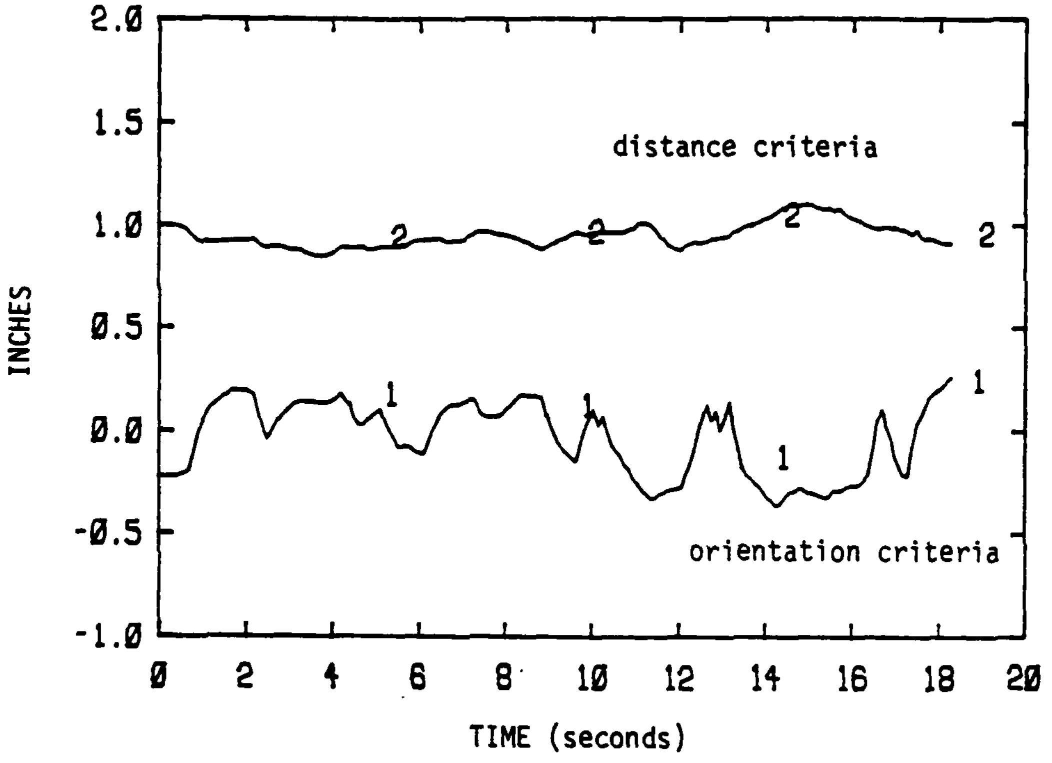


Figure 7.4 Performance display. After each run, subjects could see the orientation and distance criteria plotted versus time.

shown in figure 7.5.

The effect of control mode was found to be highly significant both for RMS distance error ($p < .005$) and for RMS orientation error ($p < .025$). The subject means for RMS error are shown in figure 7.6.

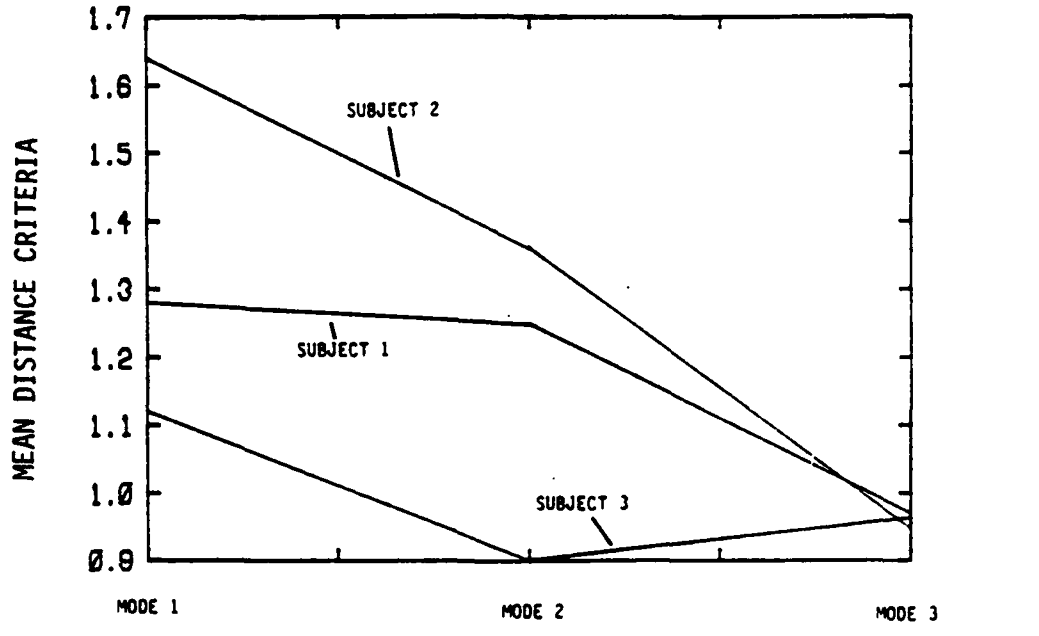
The effect of control mode on time was also highly significant ($p < .005$). As shown in figure 7.7, the interaction between subject and control mode is extremely low.

7.5 Discussion

An examination of typical trajectories for the three control modes (figure 7.8) shows why the mean errors are not significant, while the RMS errors are highly significant. Both distance and orientation errors are highly variable throughout each run for modes 1 and 2, and much less variable in mode 3. Mean error is a poor description for the performance in these cases.

For RMS orientation error, only a small improvement was seen between modes 1 and 2, while a large improvement was seen in mode 3. For RMS distance error, a more uniform improvement was seen across the three modes.

It is not surprising that performance improved as more supervisory aids were given to the human operator. The more interesting result is which



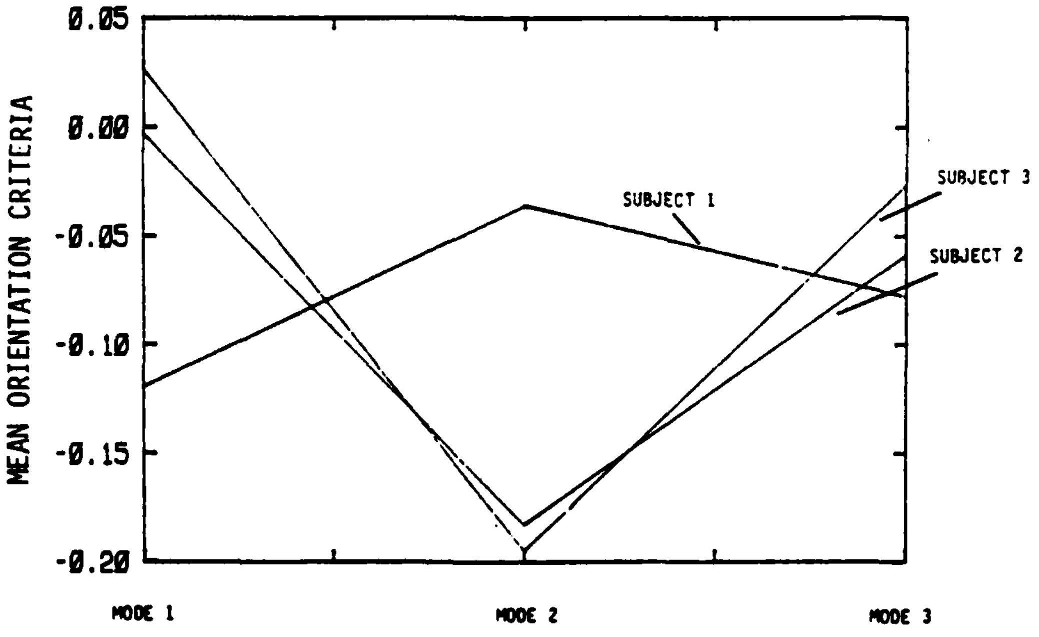
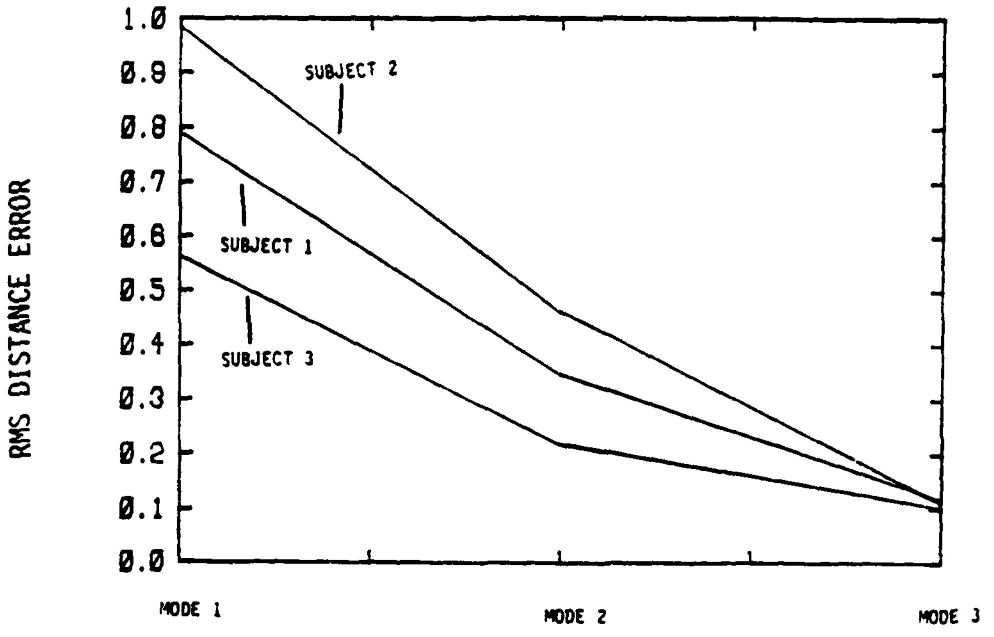


Figure 7.5 Mean performance as a function of control mode for the distance and orientation measures. No significant effects were seen.



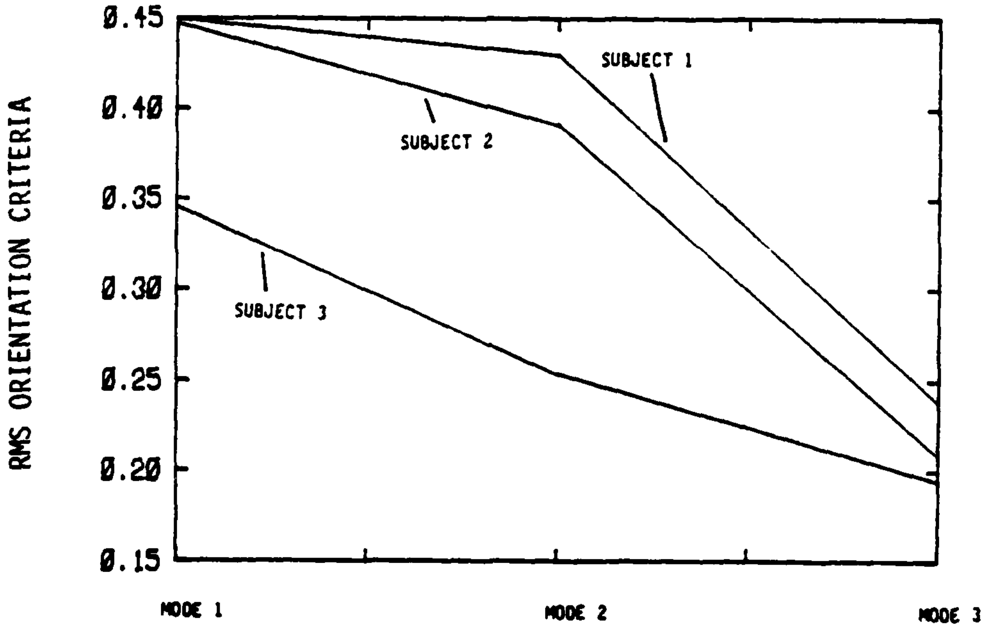


Figure 7.6 RMS performance as a function of control mode for the distance and orientation measures. The effect of control mode was significant for both measures. The variation between subjects was much less in mode 3 for both measures.

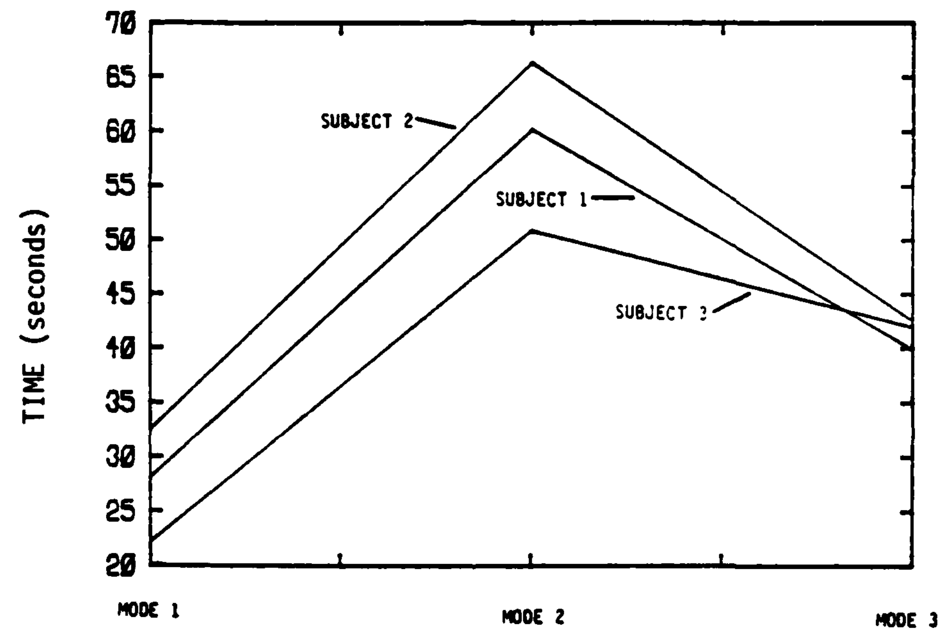


Figure 7.7 Time for each control mode. Time was lowest in mode 1. Again, the variation between subjects is lowest in mode 3.

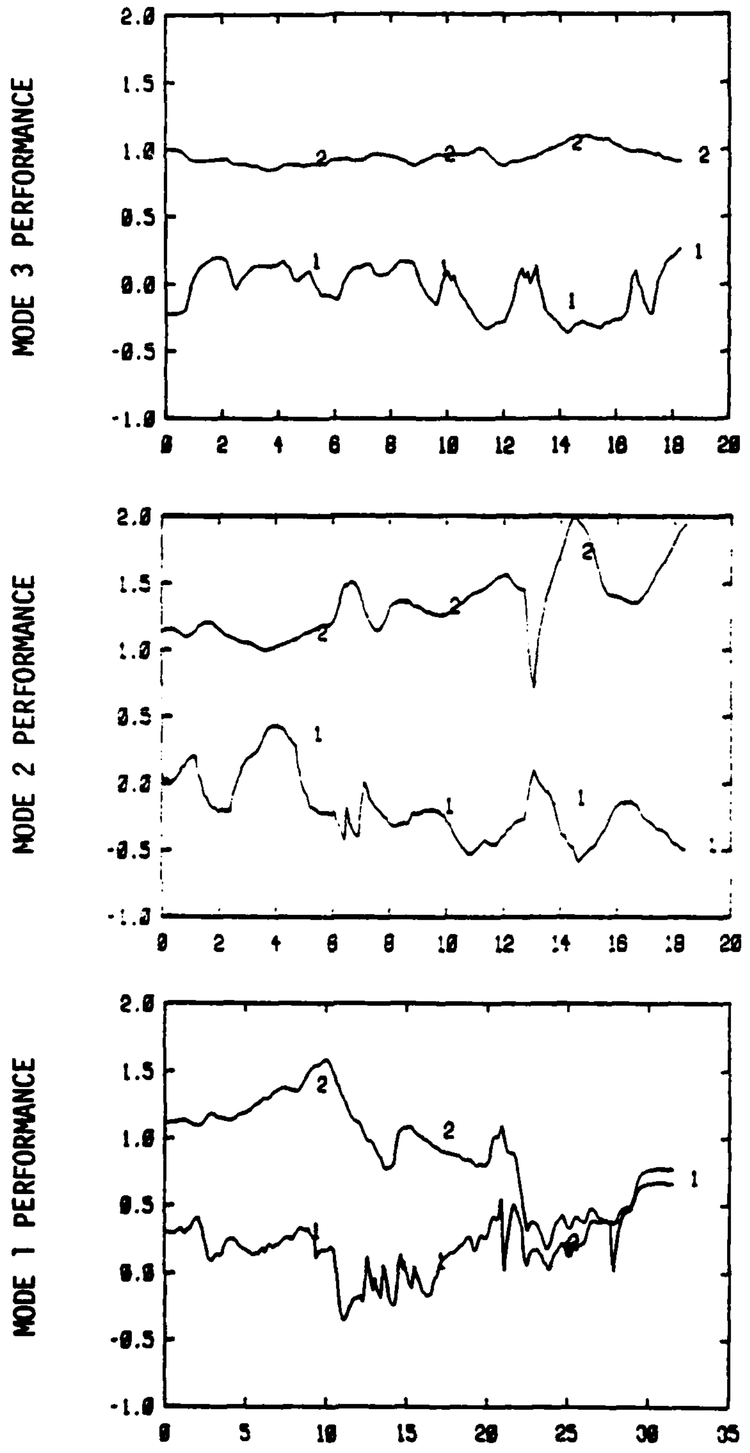


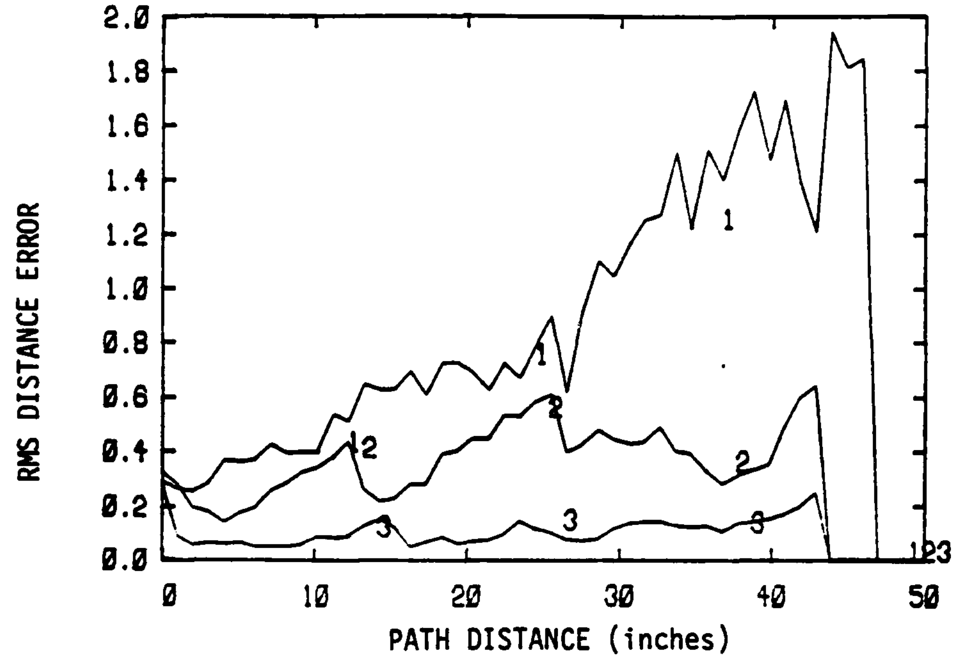
Figure 7.8 Typical performance in the three modes. For each plot, the upper curve is the distance measure, the lower curve the orientation measure. All plots are functions of time.

type of computer aids gave the best performance and how large the differences were. These results show that better performance is achieved when the person defines the object to be inspected and the computer determines the proper trajectory from that description.

The data also showed that the performance along the weld was more consistent in mode 3 over either mode 1 or mode 2. This can be seen in the plots shown in figure 7.9. These plots show RMS distance and orientation error as a function of distance along the weld across all subjects and all runs for each mode. The curve for mode 3 shows much less variability. The high variability of modes 1 and 2 is probably caused by the perceptual and motor effects which make some parts of the weld more difficult than others. The low variability in mode 3 shows that having the human teach the object to be inspected is less sensitive to the viewing conditions than either manual control or teaching the actual inspection trajectory.

The time data showed that subjects could perform the trajectory manually faster than they could teach it in either of the higher modes. This is most likely due to the fact that both kinds of teaching involved stopping and entering discrete points, while the manual control could be done continuously. The trajectories performed manually were less accurate than either of the supervisory modes.

Mode 2 showed a much larger increase in time over the manual time than did mode 3. In comparing the two supervisory control modes, not only



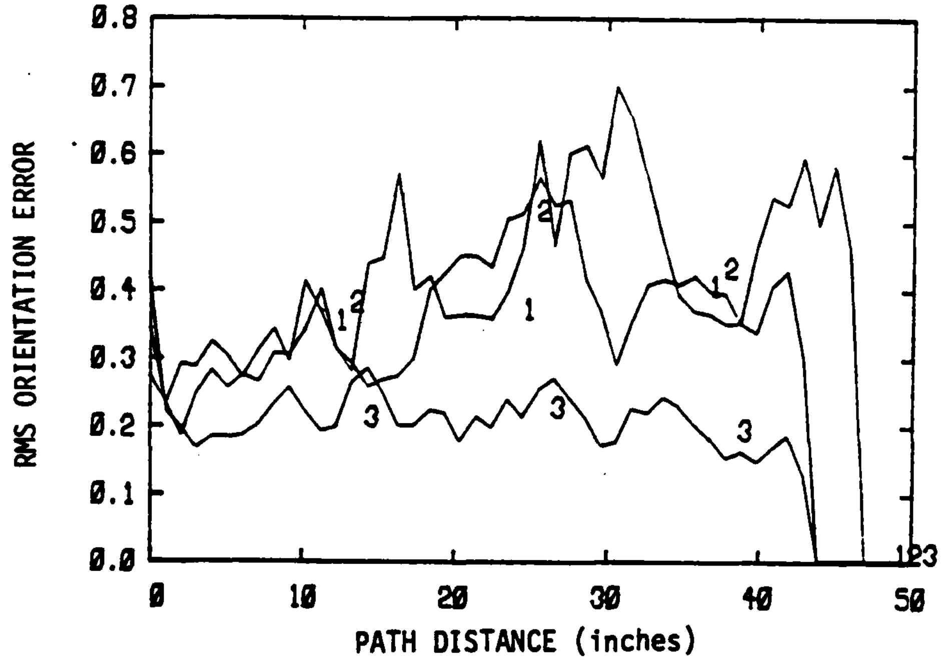


Figure 7.9 Performance as a function of distance along the weld. The upper plot is for the distance measure, and the lower plot is for the orientation measure. On each plot, curves for each mode are shown.

was it more accurate to teach the object to be inspected rather than the trajectory to follow, it was also faster. The variation between subjects was also less in mode 3.

7.6 Conclusions

This experiment compared three methods for generating trajectories such as those needed to perform a variety of inspection tasks. The experiment showed that teaching the computer by either method (mode 2 or mode 3) was more accurate that performing the task manually (mode 1).

The experiment also showed that teaching the object to be inspected to the system analogically and having the computer plan the trajectory based on the analogic information was superior to teaching the actual trajectory analogically. The performance was superior in accuracy and time. This method was also seen to be less sensitive to motor and perceptual effects. The variation between different operators was also less.

This experiment supports the design described in Chapter 4. The features of the design that were tested in the experiment were shown to increase overal system performance and decrease the dependence of the system on high quality visual feedback to the human operator.

Chapter 8

CONCLUSIONS

8.1 Design Conclusions

A supervisory underwater telemanipulation system was designed and implemented at both the MIT Man-Machine Systems Laboratory and the Naval Ocean Systems Center, San Diego. Experiments were performed that examined the specific skills that the human operator was required to perform. Another experiment looked at overall system efficacy in a simulated inspection task.

The design was derived from Sheridan's model of supervisory control [6]. The system was designed to aid the human operator in planning, teaching, monitoring, intervention, and learning. The system used an interactive interface that that two main elements:

1. A movement control language that allowed the operator to describe motions both analogically and symbolically, and combine these movement descriptions heirarchically to program tasks. The interface also allowed the operator to describe features in the environment and define operations for those features.

2. An existing computer graphic display was adapted to aid the operator in learning how the movement control language works, to help the operator develop task programs, and to aid in

monitoring the system.

The design of the system was influenced by communication with engineers from companies that build and operate remotely controlled vehicles. Because of the emphasis placed on reliability by these contacts, it was decided to try to build a system which required no increase in sensing capabilities. The system was designed to use only position feedback from the manipulator joints to the on-vehicle system and visual feedback to the human operator.

Performance improvements were sought through the introduction of computational elements which the human operator could invoke through a properly designed interface. These computational elements could be included in a real teleoperator by adding reliable, inexpensive microprocessors to the vehicle and control station, and by adding computer graphic capablilities to the control station.

8.2 Experimental Conclusions

Rassmussen [9] has proposed a hierarchical model for human behavior in control of complex systems. By this model, all tasks are comprised of knowlege-based, rule-based, and skill-based behavior. A major component of supervisory control is the off-loading of skill-based and rule-based behavior by the human operator to the computer system.

The problems associated with conducting supervisory contol experiments were discussed and a strategy was devised. For the skill-based portions of tasks which could not be off-loaded to the computer system, experiments were run which were controlled in terms of psychological factors. Then, experiments concerning overal system efficacy were run which were controlled in terms of engineering considerations.

Since the decision was made not to add sensors to the system, not all skill-based activity could be off-loaded to the computer system. In particular, the human operator was required to point out to the computer key objects or locations with the manipulator. The first experiments examined this skill-based activity. One experiment focussed on performance in pointing under direct viewing conditions, and the second focused on pointing performance when viewing through a television camera at various viewing angles.

These experiments showed that perceptual considerations dominate errors made by human operators in pointing, and that the direction of the errors is significantly correlated with viewing conditions. Errors were consistently biased toward the viewpoint, both in the direct viewing experiment and in the television viewing experiment.

The final experiment concerned overall system efficacy in a simulated inspection task. Subjects performed the task manually (mode 1) and using two forms of supervisory control. In one supervisory control mode, the subjects taught the trajectory to be followed as a series of discrete positions and orientations (mode 2). In the other supervisory control mode (mode 3), the subjects taught the object to be inspected by pointing out a series of coordinate frames, and the computer system computed the appropriate trajectory.

This experiment showed that subjects could perform the task faster manually than they could teach the task in either supervisory control mode, but the system was much more accurate using the supervisory control modes. The experiment also showed that accuracy was higher and teaching time was lower when teaching the object to be inspected rather than the actual inspection trajectory. These effects were extremely uniform across subjects. The result also implied a decreased dependence on visual feedback in mode 3.

These results show that a useful system has been built which performs better than a manually controlled system for a simulated inspection task. This test also shows the usefulness of various supervisory control features and that these features make the entire system less dependent on visual feedback.

This research verifies achievement of the design objective of increasing performance without increasing complexity or decreasing reliability by adding sensors has been achieved. At the same time, the dependence on visual feedback has been decreased.

8.3 Future Work and Recommendations

This work will be continued both at MIT and at NOSC. Tests of this system at NOSC, both in air and in water are planned.

At MIT, the conclusion that this system is less sensitive to the quality of the visual feedback to the operator will be extended to include the sampling and time delay effects of slow-scan television.

Additionally, more research on how to teach the system about objects to be inspected is planned. In the work completed, no previous knowledge of the shape of the weld was assumed. The operator taught the system about the weld to be inspected by entering a series of coordinate frames. Perhaps the teaching can be done faster by combining frames pointed out by the operator with a model of the weld.

It is the hope of the author that this work has contributed to a better understanding of how work may be done remotely. Issues of safety and productivity make progress in this area essential.

Appendix A

Inverse Kinematics for the NOSC Manipulator

The NOSC manipulator is shown in figure A-1.

The joint angles required to bring the hand to a desired hand frame may be calculated in the following fashion:

A.1 Define wrist point in base frame

$$X_w = \begin{matrix} x_w \\ y_w \\ z_w \\ 1 \end{matrix} = {}^0A_h \begin{vmatrix} 0 \\ 0 \\ -s_6 \\ 1 \end{vmatrix}$$

A.2 Solve for the first joint angle (2 possible solutions)

$$\theta_1 = \tan^{-1}(-x_w, y_w)$$

In this case, the arctan function is passed 2 arguments, and can keep the resulting angle in the proper quadrant.

A.3 Solve for the third joint angle (2 possible solutions for each value of θ_1)

This angle may be calculated using the Law of Cosines.

$$R_w^2 = x_w^2 + y_w^2 + z_w^2$$

$$\theta_3 = \frac{\pi}{2} \pm \cos^{-1}\left(\frac{a_2^2 + s_4^2 + R_w^2}{2 * a_2 * s_4} \right)$$

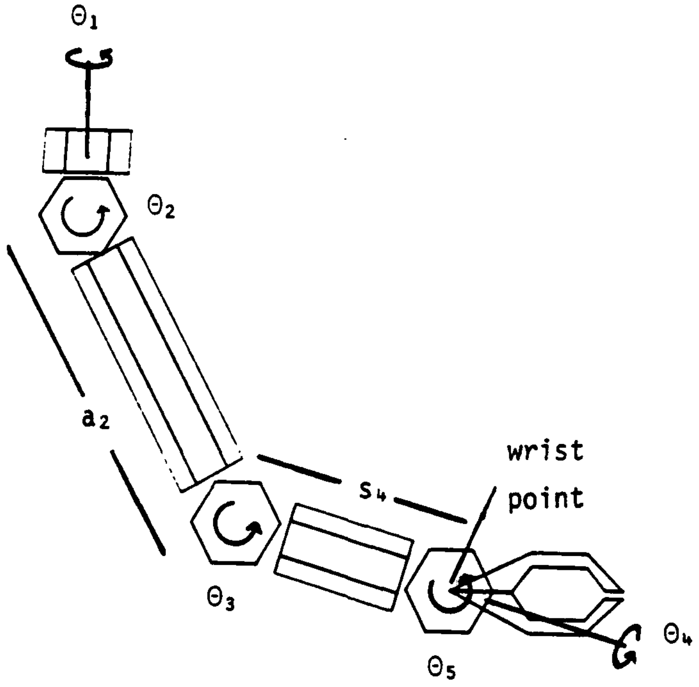


Figure A.1 The NOSC manipulator

A.4 Solve for the second joint angle (one solution)

Again, using the Law of Cosines

$$R_s^2 = x_w^2 + y_w^2$$

$$\theta_2 = \tan^{-1}(R_s, z_w) - \cos^{-1}\left(\frac{R_w^2 + a_2^2 + s_4^2}{2 * a_2 * R_w}\right)$$

A.5 Transform translation of the tip of the hand to a coordinate frame centered on the wrist that does not include 0_4.

$$X_T = \begin{vmatrix} x_T \\ y_T \\ z_T \\ 1 \end{vmatrix} = \begin{vmatrix} C_1 & S_1 & 0 & 0 \\ S_1 * C_{23} & C_1 * C_{23} & S_{23} & a_2 * S_{23} \\ S_1 * S_{23} & C_1 * S_{23} & C_{23} & a_2 * C_3 + s_4 \\ 0 & 0 & 0 & 0 \end{vmatrix} \begin{vmatrix} x_h \\ y_h \\ z_h \\ 1 \end{vmatrix}$$

$$0_5 = \tan^{-1}((\sin\theta_4 * x_T + \cos\theta_4 * y_T), - z_T)$$

Appendix B

Inverse Kinematic Solution for the Argonne E-2 Master-Slave Manipulator

B.1 Introduction

Cartesian position control of a manipulator arm requires that a solution be obtained for the manipulator joint angles, given a desired hand position and orientation expressed in cartesian form. In this appendix, a method will be presented for obtaining the joint space solution for the E-2 manipulator from the hand position and orientation expressed as a homogeneous transformation.

B.2 *Kinematic Difficulties with the E-2*

The inverse kinematics of the E-2 arm are difficult, as the arm was not designed with computer control in mind. This is not surprising, as the E-2 was designed in the early 1950's, while methods for cartesian position control were not developed until the mid-1960's [19], [21]
Arms designed for computer control are usually built with the last three axes of rotation intersecting at a point. This allows the 6 dimensional problem to be broken down into two three dimensional problems which can both be solved analytically [21]. The E-2 arm does not have this feature, as a result obtaining the joint space solution becomes much more difficult.

Manipulator kinematics are commonly expressed using notation developed by Denavit [23] and developed for manipulators by Roth and Pieper [19]. In this notation, the E-2 arm is classified as:

$$s_2 s_4 a_4$$

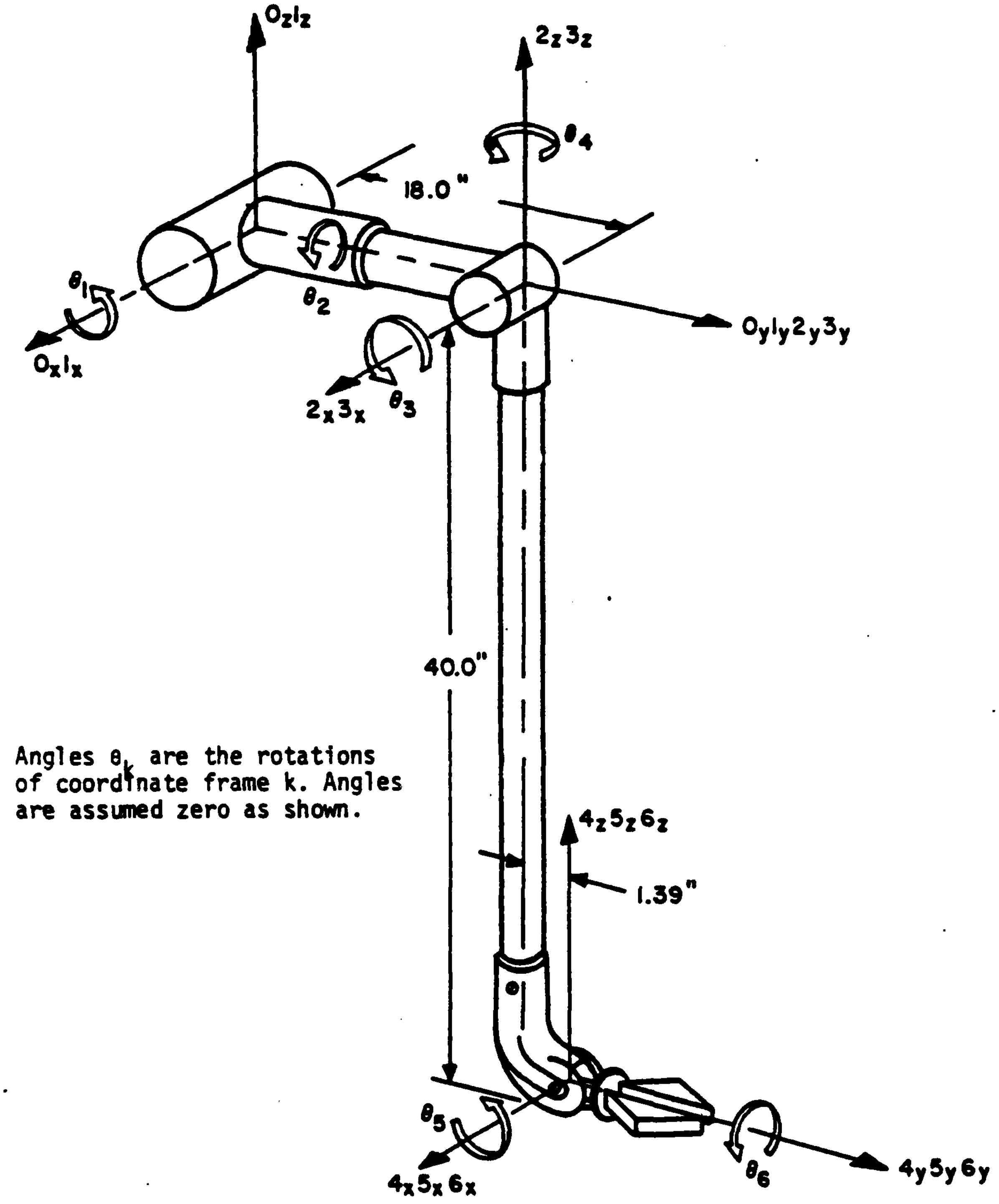
Roth [19] notes that this type of arm is solvable.

B.3 Solution Method

The inverse kinematic solution will be obtained by first solving for a similar arm for which a solution is straightforward. This solution can then be used to arrive at the proper solution for the E-2. A satisfactory approximate solution for the E-2 can be obtained using a one-step transformation, or the solution can be computed interactively if more accuracy is needed.

B.3.1 Approximate E-2 Kinematics

The E-2 may be approximated by the arm shown in figure B.2. This is similar to the actual E-2, but has the last three axes intersecting. A closed-form solution may be obtained for this arm.



Angles θ_k are the rotations of coordinate frame k. Angles are assumed zero as shown.

Figure B.1 Argonne E-2 Manipulator (from Brooks)

B.3.2 Wrist Position Solution

The cartesian position of the wrist point may be obtained directly from the homogeneous transformation for the hand, given the length of the hand.

$$\begin{vmatrix} x \\ y \\ z \\ 1 \end{vmatrix}_w = {}^0A_H \begin{vmatrix} 0 \\ -s_6 \\ 0 \\ 1 \end{vmatrix}$$

where ${}^0A_H$ is the current hand frame and s_6 is the length of the hand.

Once the wrist point position is known, the first three joint angles can be computed. The third joint angle, θ_3, may be found using the Law of Cosines.

$$r_w^2 = x_w^2 + y_w^2 + z_w^2$$

$$\theta_3 = \cos^{-1}\left(\frac{s_2^{\,2} + s_4^{\,2} - r_w^{\,2}}{2s_2s_4}\right) - \pi/2$$

If θ_3 is known, θ_2 can be obtained from the value of x_w, since x_w does not depend on θ_1.

$$\theta_2 = \sin^{-1}\frac{x_w}{-s_4 \cos\theta_3}$$

Given θ_2 and θ_3, θ_1 may be computed:

$$\theta_1 = \sin^{-1}\frac{y_w * s_4 * \cos\theta_2 * \cos\theta_3 + (s_2 + s_4 * \sin\theta_3)\, z_w}{(\cos\theta_2 * s_4 * \sin\theta_3)^2 + (s_2 + s_4 \cos\theta_3)^2}$$

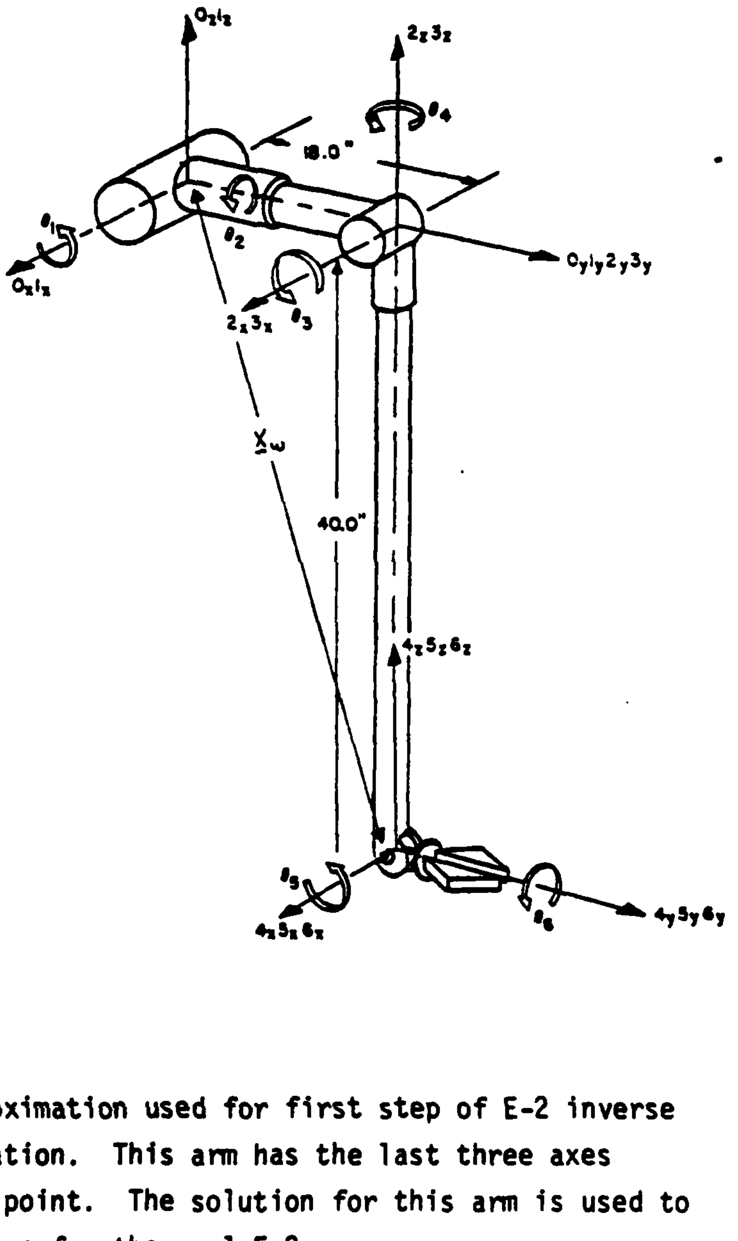


Figure B.2. Approximation used for first step of E-2 inverse kinematics computation. This arm has the last three axes intersecting at a point. The solution for this arm is used to compute the solution for the real E-2.

B.3.3 Wrist Angle Solution

After the first three joint angles have been computed, the last three angles may be computed. The transformation due to the last three angles ${}^3A_H$, may be determined from the hand frame, ${}^0A_H$, and the transformation due to the first three angles, ${}^0A_3$.

$$ {}^3A_H = ({}^0A_3)^{-1} * {}^0A_H $$

The values of the first three joint angles may now be computed given that the wrist is a yaw-pitch-roll system.

B.3.4 Transforming to the Actual E-2 Kinematics.

Once values for 0_1 through 0_4 are known for the approximate kinematics, a close approximation to the real kinematics may be obtained. First, corrections to 0_2 and 0_3 may be obtained based on the approximate values of 0_1 through 0_4.

$$ \theta_2 = -a_4 * \cos(\theta_1 + \theta_3) * \cos \theta_4/s_4 $$

$$ \theta_3 = -a_4 * \cos(\theta_1 + \theta_3) * \sin \theta_4/s_4 $$

Then, the last three angles can be computed based on the corrected values for θ_2 and θ_3.

Appendix C

Servo Control of the NOSC Manipulator

C.1 Introduction

This appendix describes the direct digital servo routine (ARM) which continuously control the position of each rotary joint of the NOSC manipulator. This program receives discrete joint space commands from the supervisory command processor which computes the joint space positions from high level specifications. The ARM program generates a continuous position trajectory from the starting joint positions to the final joint positions, and generates actuator command signals to keep the joints on the prescribed trajectory.

The performance of these servoes in position control is good. No position overshoot is seen and steady state error approaches the resolution of the A/D cor [illegible]ter.

C.2 Dynamic Control

The system allows the position and velocity of the joints to be controlled. High level cartesian transformations are performed by the Supervisory Command Processor. The ARM program works only in joint space.

Each joint consists of a high performance DC torque motor, a harmonic drive reduction mechanism, and a potentiometer which measures the position of the joint (not the motor). A block diagram is shown in figure C-1. The motor is voltage controlled through a linear amplifier. The inputs to the linear amplifiers are D/A converter outputs from an LSI 11/23 microcomputer. The voltage from the potentiometer is read by an A/D on the LSI.

The control system regulates each joint individually, although scheduled gains are used on the first two joints to compensate for changing inertia as the arm extends. Because of the high gear reductions used, the inertia of the motor and the high speed end of the harmonic drive mechanism tend to dominate. The dynamic and static effects of changes in configuration are less than would be expected. An expression for the effective inertia of each joint as seen in joint coordinates:

$$J_e = n^2 J_m + J_a$$

where J_e is the effective inertia

J_m is the inertia of the motor and high speed end of the gear drive

J_a is the inertia of the arm about rotation of the joint

n^2 is the gear ratio

For the first joint with the arm half extended, these values are:

$$J_e = 106 \text{ oz-in-sec}$$
$$n^2 J_m = 50 \text{ oz-in-sec}$$
$$J_a = 56 \text{ on-in-sec}$$

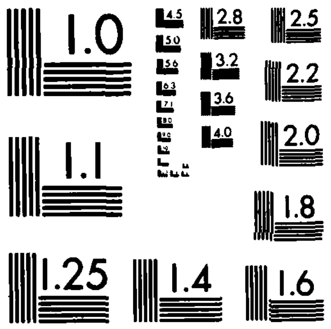
AD-A123 632 SUPERVISORY CONTROL OF UNDERWATER TELEMANIPULATORS: DESIGN AND EXPERIMENT(U) MASSACHUSETTS INST OF TECH CAMBRIDGE MAN-MACHINE SYSTEMS LAB D R YOERGER 30 AUG 82 N00014-77-C-0256 3/3

UNCLASSIFIED F/G 5/8 NL

END

FILMED

DTIC


1.0
4.5
5.0
5.6
6.3
7.1
8.0
9.0
2.8
3.2
3.6
4.0
2.5
2.2
2.0
1.1
1.8
1.25
1.4
1.6

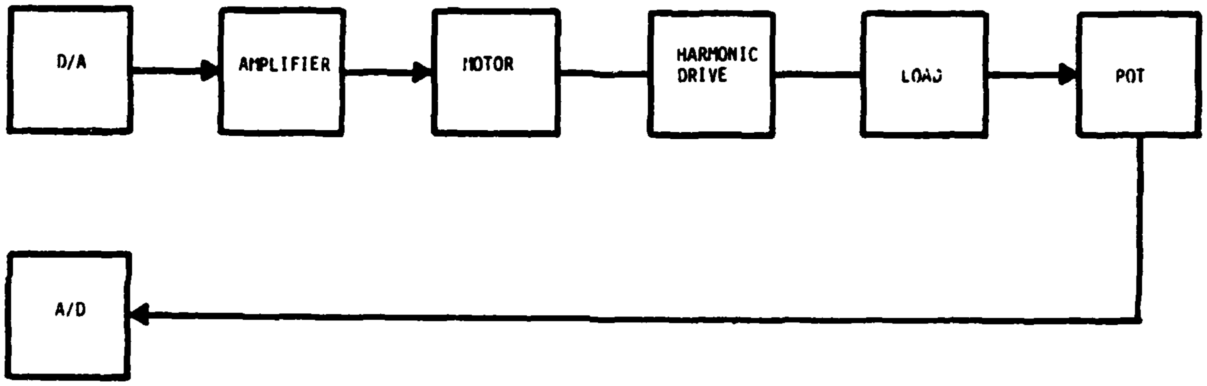


Figure C.1. Block diagram of a single degree of freedom.

Changes in J are less important in this case, as the total change in effective inertia is not so large.

In summary, the design approach will be to control each joint separately, although scheduled gains will be used on the first two joints. Scheduled gains compensate for the static effects of changes in joint configuration (changes in inertia and gravity effects). Dynamic effects resulting from changing configuration, such as coriolis and centripedal forces, will be compensated with feedback.

Each joint has a proportional position controller with both setpoint and voltage output limits, as shown in figure C.2.

These controllers each have four parameters. the following list shows the parameters as shown in figure C.2 and the name by which they are called in both the ARM program and in the supervisory command processor (communication between these two programs will be discussed in the next section).

K_p	Forward gain (PGAIN)
θ_u ,θ_l	Upper and lower position setpoint limits (MXSTP, MNSTP)
V_l	Voltage limit (LIM)

These parameters were adjusted empirically for good transient response and low steady state error.

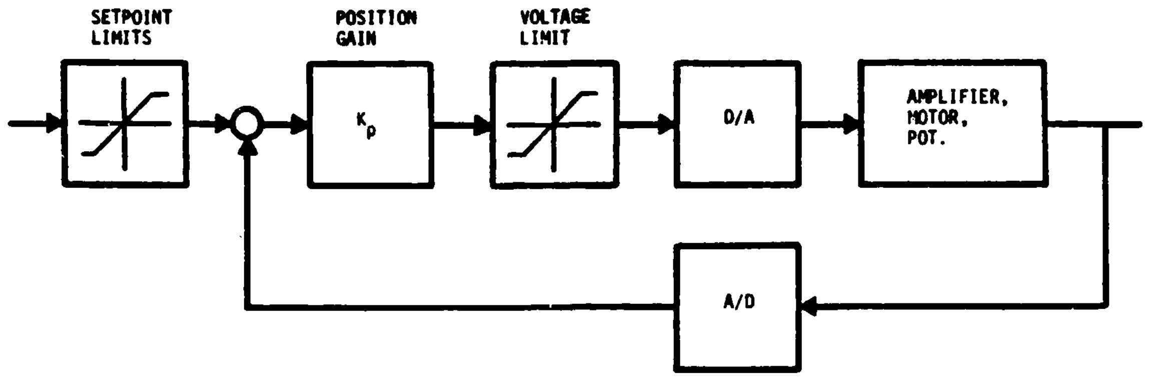


Figure C.2. Position compensator.

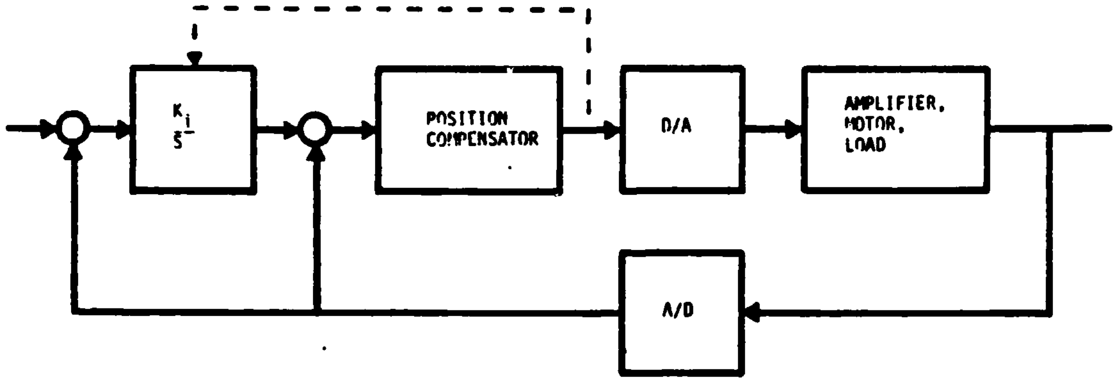


Figure C.3. Integral controller.

Joint 2 must lift the arm against gravity, so an additional integral loop was closed around the proportional position loop, as shown in figure C.3. The integral controller was the so-called "intelligent integrator" which prevents integrator windup when the proportional controller saturates. This controller had one additional parameter, the integral gain (IGAIN). The controller achieved steady state position errors near the limit of resolution of the A/D converter despite the large gravity load.

Scheduled gains were used on joints 1 and 2, those most effected by changing inertias. Again, the gains were chosen empirically. First, gains were chosen with the arm fully extended, then gains were chosen with the arm retracted. In real-time, the gains were changed between these two values by linearly interpolating with respect to distance the arm was extended.

C.3 Communication with the Supervisory Command Processor.

The user of this system does not need to be concerned with communication between the two programs which control the arm. However, an understanding of this communication is needed for engineers working on the system.

All parameters of the controllers may be read or changed from the supervisory command processor. To look at any of the values, the user inputs:

```
                  n name DISPLAY
            where    n is the joint number (1-6)
                     name is the name of the variable to be
                     displayed (i.e. LIM, PGAIN, etc.)
            example: 1 LIM DISPLAY (types out the value of the
                     voltage limit for joint 1)
```

To change any value, the user inputs:

```
                  value n name CHANGE
            where      value is the new value of the parameter
                       n is the joint number
                       name is the name of the parameter to be
                       changed
            example:   1000 1 LIM CHANGE (changes the voltage
                       limit for joint 1 to 1000)
```

Setpoints for the controllers are received from the supervisory command processor through a ring buffer. The ARM program gets joint vectors from the buffer as they are to be executed, generates a linear joint space trajectory to that joint vector while servoing the joints, and then gets another vector from the buffer, until the buffer is emptied.

Appendix D

Cartesian manipulator control language

Data types

INTEGER	16 bit integer
JFRAME	6 dimensional vector of joint angles, 32 bit floating point representation
CFRAME	4 x 4 homogeneous cartesian transform, 32 bit floating point numbers
VECTOR	3 dimensional vector, 32 bit floating point numbers

Kinematics

JTOC	compute forward kinematics
CTOJ	compute inverse kinematics

Transformations

CMUL	multiply two homogeneous transforms
CTOV	extract the translational vector portion of a homogeneous transform
CREL	compute the relative transform between two homogeneous transforms
VTOC	compute a cartesian transform from three vectors

Sensing

?TOUCH	returns state of touch sensor
CFORCE	returns estimates of hand forces resolved in the hand frame
CIAD	returns a homogeneous transform corresponding to the current manipulator position
JIAD	returns a joint vector corresponding to the current manipulator position

Motions

JMOVE	move the arm to the position and orientation specified by a vector of joint angles
CMOVE	move the arm to the position and orientation specified by a homogenous transform using the base coordinate frame as a reference
CREL	similar to CMOVE, except that the current hand frame is used as a reference
VREL	translates the manipulator hand relative to the current hand frame, the translation is specified by a VECTOR

Appendix E

Formal description of movement control language

E.1 Formal Descriptions

A formalism is needed to precisely specify a language. Computer languages are commonly specified using Backus-Naur Form (BNF) [DONOVAN]. In BNF, an entity is specified in terms of other defined entities and possibly itself. BNF descriptions are an unambiguous way of specifying the syntax of a language.

Example:
```
<letter> ::= A | B | C | ...   Z

<digit> ::= 0  | 1 | 2 | ...   9

<character> ::= <digit>   <letter> | ! | @ | # | $ | % | ©
                & | * | ( | ) | _ | - | + | = | ™ | ° | §
                † | [ | ] | : | ; | " | ' | ® | < | > | ,
                . | ? | /

<integer> ::= [<digit>] | - <integer>

<name> ::=[<character>]
```
The pointy brackets indicate families, so <digit> means " the family of digits". The symbol " ::= " means " is replaced by ". the vertical bar " | " means " or " . Square brackets " [ ] " mean " any combination of " .

This example states that a letter consists of any one of the symbols A-Z, digits consist of any of the symbols 0-9. A character consists of any digit, letter, or any of the other symbols listed. An integer consists of any combination of digits, possibly preceded by a minus sign. A name is made up of a combination of characters.

E.2 Structures for defining coordinate frames analogically

E.2.1 Absolute coordinate frames

A single absolute coordinate frame may be represented by a structure called a POSITION. A POSITION is established by giving the command

POSITION <name>.

The system will go into a preselected manual control mode. The operator can then move the arm to the desired position and orientation. The position and orientation indicated are then stored as a homogeneous transform under the name given. This results in the creation of a new member of the POSITION family. For the purpose of formal definitions, any member of the POSITION family may be referred to as

<position>

Any number of absolute frames can be established as a PATH. Each frame in a PATH is entered in the same way as a POSITION. The procedure of establishing a PATH is begun by giving the command

PATH <name>.

An individual coordinate frame in a PATH may be referred to as:

<integer> <path>

where the integer refers to which element of the path is desired.

These two methods may be summarized formally:

<absolute frame> ::= <position> | <integer> <path>

E.2.2 Relative Coordinate Frames

The first structure is called a REL_FRAME. When the operator gives the command

REL_FRAME <name>,

the current hand frame is recorded, then the system is placed in manual control. The operator then indicates another position and orientation. The system records the homogeneous transform that would transform the original frame into the final frame under the name given.

The structure REL_PATH allows a series of frames defined relative to an arbitrary reference to be recorded. When the command

REL_PATH <name>

is given, the current hand frame is recorded, and the operator may then indicate any number of frames. These frames are recorded as relative transformations. Any of these relative frames may be referred to by:

<integer> <rel_path>.

Relative coordinate frames may be summarized formally:

<relative frame> ::= <rel_frame> <integer> <rel_path>

E.2.3 Displaying Analogically Defined Coordinate Frames

All frames may be shown on the graphic display after they have been defined using the SHOW command. Absolute frames are shown relative to the base frame, while relative frames are shown relative to the current hand frame. <display> ::= <absolute frame> SHOW|<relative frame> SHOW

E.3 Defining motions

Motions may be programmed either by using any analogic structures th have been created, or motions may be defined purely symbolically.

E.3.1 Relative motions

The MOVE command may be preceded by any properly defined relative frame. This may be stated formally as:

<relative frame> MOVE

Relative motions may also be defined symbolically. Using symbolic relative motion commands, it is possible to make the arm translate or rotate relative to the current hand frame.

For relative translations, the proper syntax is:

```
<relative translation> ::= <integer> <unit> <direction> |
                           <integer> <unit> |
                           <integer> <unit>
```

```
<integer> <unit> TRANSLATE
```

Acceptable units are:

```
<unit> ::= MM | CM | INCHES | FEET
```

Directions are:

```
<direction> ::= UP    DOWN    LEFT    RIGHT    FORWARD    BACK
```

These directions are shown in figure 4.10a. For the TRANSLATE command, the three arguments are x, y, and z distance.

For relative rotations, the proper syntax is:

```
<relative rotation> ::= <integer> <angle>
```

where angles are defined by:

```
<angle> ::= PITCH | ROLL | YAW
```

The integer argument is stated in degrees.

Examples

```
45 ROLL
-30 YAW
```

Relative motions may be summarized formally:

```
<relative motion> ::= <relative frame> MOVE   |
                      <relative translation>  |
                      <relative rotation>
```

E.3.2 Absolute motions

The first type of absolute motion has the syntax

```
<absolute coordinate> MOVE.
```

This command causes the arm to move from its current position and orientation to any properly defined absolute coordinate frame.

The second type of absolute motion is used to move the arm to a position or orientation defined relative to an absolute coordinate frame. the proper syntax is:

```
<absolute coordinate> SET <relative motion>
```

E.3.3 Relative Trajectories

Relative trajectories may consist of a series of symbolic translations or rotations. Another useful command is the TRAVERSE command which causes the sequence of relative motions stored as a REL_PATH to be executed. The syntax of this command is:

```
<rel_path> TRAVERSE
```

Relative trajectories are formally defined as:

```
<relative trajectory> ::=
                    <relative movement> |
                    <rel_path> TRAVERSE |
                    <relative trajectory> <relative movement>
```

E.3.4 Absolute Trajectories

Absolute trajectories are sequences of movements which begin with an absolute movement. The TRAVERSE command also works with PATHs. The

REVERSE command moves the arm through all the frames of a PATH in reverse order.

Absolute trajectories are formally defined as:

```
<absolute trajectory> ::=
                    <absolute movement> |
                    <path> TRAVERSE |
                    <path> REVERSE |
                    <absolute trajectory> <relative trajectory> |
                    [<absolute trajectory>]
```

E.4 Sensing commands

The three sensing commands may be summarized formally:

```
                <sensor> ::= ?TOUCH | ?FORCE | ?TORQUE
```

E.5 Building manipulation tasks using movements

E.5.1 Extensibility

Extensibility using both the colon and the DEFINE: commands may be formally summarized:

```
<trajectory> ::= <absolute trajectory> | <relative trajectory>
<command> ::= [<trajectory>]   <display> |
              DEFINE: <name> [<command>] ; |
              : <name> [<command>] ;
```

E.5.2 Decisions

Programs may be written which branch depending on the sensing commands. Structures for this include DO...LOOP, IF...ELSE...ENDIF, and BEGIN...UNTIL. These structures are used as follows:

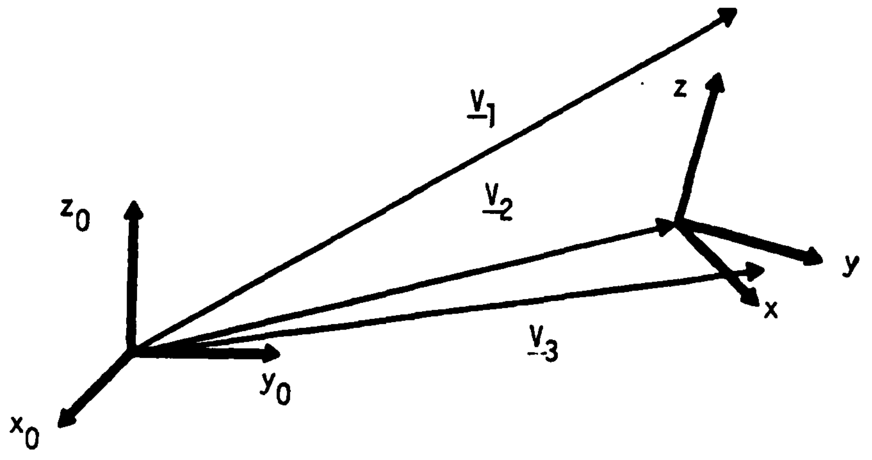
```
<integer> TIMES DO  [<command>]  LOOP
<sensor> IF [<command>] ELSE [<command>] ENDIF
BEGIN [<command>] <sensor> UNTIL
```

Each of these structures must appear inside colon or DEFINE: definitions.

Appendix F

Computation of a Coordinate Frame from Three Vectors

Given 3 vectors V_1, $\underline{V}_2$, and $\underline{V}_3$



1. The z axis may be computed by subtracting $\underline{V}_2$ from $\underline{V}_1$, and normalizing the vector

$$\underline{Z} = \frac{\underline{V}_1 - \underline{V}_2}{\underline{V}_1 - \underline{V}_2} \qquad \frac{\underline{V}_1 - \underline{V}_2}{\underline{V}_1 - \underline{V}_2}$$

2. The x axis may be formed by first computing $\underline{V}_3 - \underline{V}_1$, and computing the cross product of this vector and $\underline{Z}$

$$\underline{X} = \underline{Z} \times \frac{\underline{V}_3 - \underline{V}_1}{\underline{V}_3 - \underline{V}_1}$$

3. The y axis is formed by completing the right handed set

$$\underline{y} = \underline{z} \times \underline{x}$$

4. The cartesian transform may be formed by placing the three vectors which describe the axes in the rotation part of the transform, and setting the translation part of the transform equal to $\underline{V}_2$

$$A = \begin{vmatrix} x_1 & y_1 & z_1 & V_{2x} \\ x_2 & y_2 & z_2 & V_{2y} \\ x_3 & y_3 & z_3 & V_{3z} \\ 0 & 0 & 0 & 1 \end{vmatrix}$$

APPENDIX G

A Review of Underwater Inspection and Maintainence Tasks

This appendix reviews a series of underwater tasks which are being investigated at the Norwegian Underwater Technology Center (NUTEC) in conjuction with the Central Research Institute (SI). The effort at NUTEC is focused on building an integrated system consisting of a manipulator and control system, tools, and a man-machine interface. The tasks which they have selected represent tasks which would be useful to the commercial offshore community and which may be practical to perform remotely.

At NUTEC during 1981, a total of 165 hours of testing in the water have been performed on the following tasks:

*Clean steel with a water jet

*Clean steel by sand washing

*Piling bricks

*Screw nut on and off with an impact wrench

*Mount and dismount a flange

*Take cathodic protection readings

*Take ultrasound readings

*Clean a weld with a brush

Cut chain with a grinder

Cut wire with a ram

*Mount and remove a bolt

*Open and close a valve

Mount and dismount a hose coupling

Attach a snap hook

*Measurement

*Grind a weld

The tasks in this list which are marked by an asterix are those where the system presented in this thesis could potentially increase performance. The inspection task used in the experiment presented in chapter 7 was patterned after the jet and sand cleaning task and the weld grinding task noted above.

Appendix H

Kinematic Notation

Manipulator kinematics are commonly classified using notation originally designed by Hartenberg and Denavit, and developed for manipulators by Roth and Pieper. This notation will be outlined for arms with rotational joints in this appendix.

A coordinate system is attached to each joint so that the acis of rotation of that joint is the z axis of the coordinate system. The notation completely describes the relationship between these coordinate systems, and therefore the kinematics of the arm.

Each coordinate system is defined in the following way, as described by Hollerbach:

z_i is directed along the axis of joint $i + 1$

x_i lies along the common normal from z_{i-1} to z_i

y_i completes the right handed coordinate system

As shown in figure H.1, the relationship between any two coordinate frames may be established by 4 parameters:

a_i distance between the origins of coordinate systems i-1 and i measured along x_i.

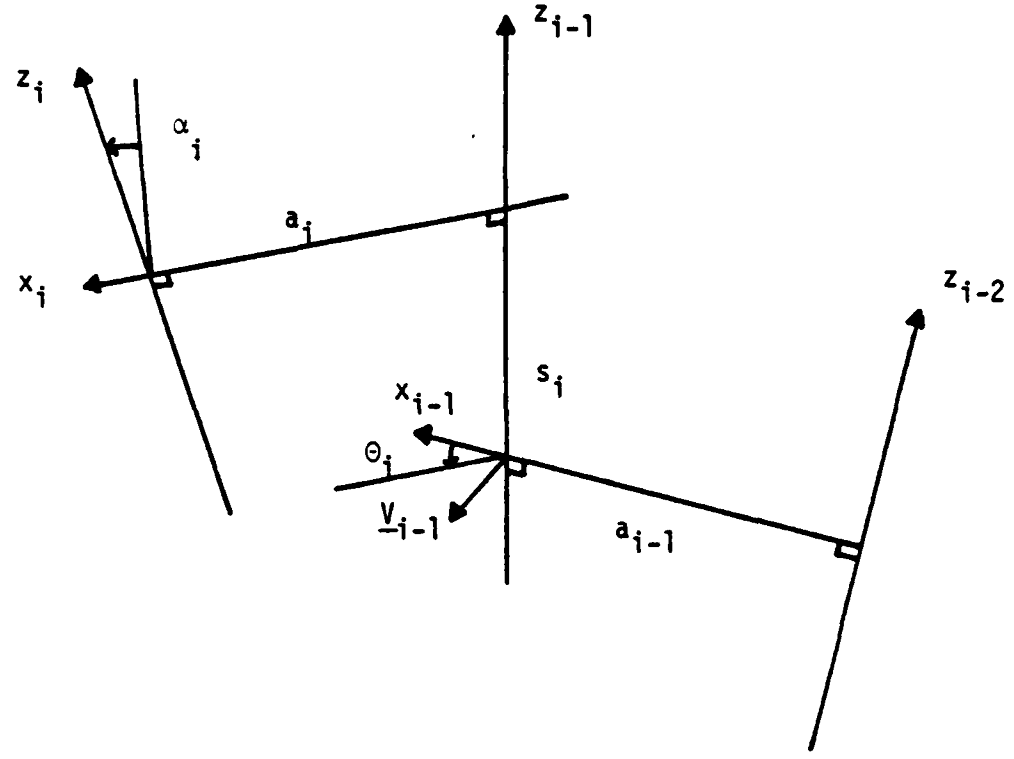


Figure H.1. Relationship between coordinate frames for specifying kinematics of a manipulator.

s_i distance between x_{i-1} and x_i measured along z_{i-1}.

α_i angle between the z_{i-1} and z_i axes measured in a right handed sense about x_i.

θ_i angle between the x_{i-1} and x_i axes measured in the right-handed sense about z_{i-1}. This is the angle of rotation for a rotary joint.

Manipulators are commonly classified in terms of the values of a and s. The angle θ describes the specific configuration of the arm, and the angle α doesn't influence the solvability of the arm. For a well designed arm, only a few of the a and s parameters will be non-zero.

For the NOSC arm, the upper arm link is a_2 and the lower arm link is s_4, so the arm is classified as a_2s_4.

For the E-2 manipulator, the upper arm link is s_2, the lower arm link is s_4, and the offset in the wrist is a_4. The E-2 is classified as $s_2a_4s_4$.

REFERENCES

1. Sheridan, T. B., Verplank, W. L.
"Human and Computer Control of Undersea Teleoperators", Man-Machine Systems Laboratory Report, MIT Dept. of Mech. Eng., July, 1978.

2. Sofyanos, T.N.
"An Assessment of Undersea Teleoperators", S.M. Thesis, MIT Dept. of Mech. Eng., Cambridge, 1980.

3. Busby, F.R. and Associates
"Remotely Operated Vehicles," US Dept. of Commerce, No. 03-78-G03-0136, August, 1979.

4. Linstad, A., Fondevik, W.
"Underwater Work System test at NUTEC", Report No. 1 (English Edition), Norwegian Underwater Technical Center, No. 32-82, Bergen, Norway, 1981.

5. Bertsche, W.R., Logan, K.P., Pesch, A.J., Wernli, R.L.
"Evaluation of the Design and Undersea Work Capability of the Work Systems Package", Naval Ocean Systems Center Technical Report 214, San Diego, April, 1978.

6. Rasmussen, J.
"Outlines of a Hybrid Model of the Process Plant Operator", in Sheridan, T. and Johannsen, G., eds. Monitoring Behavior and Supervisory Control, Plenum, New York 1976.

7. Whitney, D.
"Resolved Motion Rate Control of Resolved Manipulators and Prostheses", Fifth Annual NASA-University Conference on Manual Control, MIT, Cambridge, MA, March 1969.

8. Sheridan, T.B ., Ferrell, W.R.
"Man-Machine Systems: Information, Control, and Decision Models of Human Performance", MIT Press, Cambridge, 1981 edition.

9. Rasmussen, J.
"Outlines of Hybrid Model of the Process Plant Operator", in Sheridan, T. and Johannsen, G., eds. Monitoring Behavior and Supervisory Control, Plenum, New York 1976.

10. Brooks, T.L.
"SUPERMAN: A System for Supervisory Manipulation and the Study of Human/Computer Interactions", SM Thesis, MIT Dept. of Mech. Eng., May 1979.

11. Crooks, W.H., Shaket, E., Alperovitch, Y.
"Man-Machine Communication in Computer-Aided Remote Manipulation," Perceptronics, Tech. Report PATR-1034-78-3, Woodland Hills, Ca., March, 1978.

12. Crooks, W.H., Shaket, E., Chu, Y.-Y., Alperovitch, Y.
"Man-Machine Communication in Computer-Aided Remote Manipulation," Perceptronics, Tech. Report PATR-1034-79-3, Woodland Hills, Ca., March, 1979.

13. Groome, R.C.
"Force Feedback Steering of a Teleoperator System", M.S. Thesis, MIT, Dept. of Mech. Eng., August 1972.

14. Whitney, D.E.
"Force Feedback Control of Manipulator Fine Motions", ASME J. Dynamic Systems, Measurement, and Control, June, 1977.

15. Shimano, B.E.
The Kinematic Design and Force Control of Computer Controlled Manipulators, PhD Thesis, Computer Science Dept., Stanford Univ., 1978.

16. Robot Sensor Systems, Inc.
"Smart Sensors for Today's Robots", FT-1500 Force/Torque Sensor data and information brochure, 10529 Freer St, Temple City, CA, 91780.

17. Hogan, N.
"Impedance Control of a Robotic Manipulator", ASME Winter Annual Meeting, Nov, 1981.

18. Cotter, S.
SM Thesis, in progress, MIT Dept. of Mech. Eng., August, 1982.

19. Roth, B.
"Performance Evaluation of Manipulators From a Kinematic Viewpoint," NBS Special Publication, No. 459, October, 1976, pp. 39-61.

20. Mason, M.T.
"Compliance and Force Control for Computer Controlled Manipulators", IEEE J. Systems, Man, and Cyber., Vol. SMC-11, No. 6, June 1981, pp. 418-432.

21. Pieper, D.L.,
The Kinematics of Manipulators Under Computer Control, Ph.D. Thesis, Stanford Univ., 1968.

22. Roth, B.
"Performance Evaluation of Manipulators From a Kinematic Viewpoint,"
NBS Special Publication, No. 459, October, 1976, pp. 39-61.

23. Hartenberg, R.S., Denavit, J.
Kinematic Synthesis of Linkages, McGraw-Hill, 1964.

24. Woodin, A., Whitney, D.
First Annual Report for the Development of Multi-moded Remote
Manipulator Systems, C. S. Draper Lab. Report, 1972.

25. Uicker, J.J., Jr., Denavit, J., Hartenbert, R. S.
"An Iterative Method for the Displacement Analysis of Spatial
Mechanisms", J. Appl. Mech., Trans. ASME 86, 309-314, 1969.

26. Taylor, R.H.
"Planning and Execution of Straight Line Manipulator Trajectories,",
IBM J. Res. Develop, Vol. 23, July 4, 1979, pp. 424-436.

27. Bosse, P., Heckman, P.J., Jr.
"Development of an Underwater Manipulator for Use on a Free-Swiming
Unmanned Submersible", Naval Ocean Systems Center, TR 632, San Diego,
October 1980.

28. Hollerbach, J.M.
"A Recursive Lagrangian Formulation of Manipulator Dynamics and a
Comparative Study of Dynamics Formulation Complexity," IEEE J. Sys.,
Man, and Cyber., Vol. SMC-10, November 11, 1980, pp. 730-736.

29. Bejczy, A.K.
"Robot Arm Dynamics and Control", NASA-JPL Technical Memorandum,
33-669, Feb. 1974.

30. Paul, R.
"Modelling, Trajectory Control, and Servoing of a Computer-Controlled
Arm", PhD. thesis, Stanford Univ., Nov. 1972.

31. Asada, A., Kanade, T, Takeyama, I.,
"Control of a Direct Drive Arm", Carnegie-Mellon Univ.,
CMU-RI-TR-82-4, March, 1982.

32. Luh, J.Y.S., Walker, M.W., Paul, R.P.C.
"On-Line Computational Scheme for Mechanical Manipulators," J. Dynamic
Systems, Measurement, and Control, Vol. 102, June, 1980, pp. 69-76.

33. Silver, W.M.
"On the Representation of Angular Velocity and Its Effect on the
Efficiency of Manipulator Dynamics Computation", MIT AI Memo 622,
March, 1981.

34. Dubowsky, S., DesForges, D.T.
"The Application of Model-Referenced Adaptive Control to Robotic Manipulators", J. Dyn. Sys, Measurement, and Control, Sept. 1979.

35. LeBorgne, M., Ibarra, J.M., Espiau, B.
"Commande Autoadaptive de Robots Manipulateurs", (in English) Proc. of 11th ISIR, Tokyo, 1981.

36. Ogata, K.
"Modern Control Engineering", Prentice-Hall, Electrical Engineering Series, 1970.

37. Paul, R.
"Manipulator Path Control", Proc. 1975 Int. Conf. on Cybernetics and Society, IEEE, New York, Sept. 1975.

38. Luh, J.Y.S., Walker, M.W., Paul, R.P.C.
"Resolved-Acceleration Control of Mechanical Manipulators," IEEE Trans. of Auto. Control, Vol. 3, June 25, 1980, pp. 468-474.

39. Grossman, D.D.
"Programming a Computer Controlled Manipulator by Guiding Through the Motions," IBM T. J. Watson Research Center, March, 1977.

40. Gossard, D.C.
"Analogic Part Programming with Interactive Graphics", Ph.D. Thesis, MIT, Dept. of Mech. Eng., 1975.

41. Unimation Inc.
Technical brochure on the Apprentice robot, Unimation, Inc., Danbury, Ct., 1982.

42. C.I.R.P., Int. Inst. of Production Engineering Research
"Results of Survey of Current Status and Current Research and Development of Computer Automation of Manufacturing", May, 1976.

43. Whitney, D.E., Watson, P.C., Drake, S.H., Simunovic, S.
"Robot and Manipulator Control by Exteroceptive Sensors," IEEE, Joint Automatic Control Conference, 1977, pp. 155-162.

44. Sherrington, C.S.
The Integrative Action of the Nervous System, London: Constable, 1906.

45. Geldard, F.A.
"The Human Senses," John Wiley Sons, Inc., New York, 1953.

46. Sheridan, T.B ., Ferrell, W.R.
"Man-Machine Systems: Information, Control, and Decision Models of Human Performance", MIT Press, Cambridge, 1981 edition.

47. Hardin, P.A.
"AND Tree Computer Data Structures for Supervisory Control of Manipulation", Ph.D. Thesis, MIT, Dept. of Mech. Eng., 1970.

48. Sacerdoti, E.D.
"Planning in a Hierarchy of Abstraction Spaces", Third International Joint Conference on AI, 1973, pp. 412-422.

49. Crandall, S.H. (editor)
"Dynamics of Mechanical and Electromechanical Systems", McGraw-Hill, New York, 1968.

50. Thomson, W., Tait, P.G.,
Treatise on Natural Philosophy, Cambridge Univ. Press, 1879.

51. Hamilton, W.R.
"Elements of Quaternions", Third Edition, Chelsea Publishing Co., New York, 1969.

52. Bottema, O., Roth,B.
"Theoretical Kinematics", North-Holland, Amsterdam, New York, 1979.

53. McAulay, A.
"Octonions, A Development of Clifford's Bi-Quaternions", University Press, Cambridge, 1898.

54. Yang, A.T., Freudenstein, F.
"Application of Dual-Number Quarternion Algebra to the Analysis of Spatial Mechanisms," J. Appl. Mech., June, 1964, pp. 300-308.

55. Shaket, E., Freedy, A.
"A Model and Language Design for Man/Machine Communication in Computer Aided Manipulation," IEEE, Conf. on Decision and Control, 1977, pp. 553-559.

56. Grossman, D.D., Taylor, R.H.
"Interactive Generation of Object Models with a Manipulator," IEEE J. Sys., Man, and Cyber., Vol. SMC-8, Sept. 9, 1978, pp. 667-679.

57. Paul, R., Luh, J., et. al.
"Advanced Industrial Robot Control Systems", NSF Technical Report, TR-EE 80-29, Purdue Univ., July, 1980.

58. Ambler, A.P., Popplestone, R.J.
"Inferring the Positions of Bodies from Specified Spatial Relationships," Artificial Intelligence, Vol. 6, June, 1975, pp. 157-174.

59. Taylor, R.H.
"The Synthesis of Manipulator Control Programs from Task-Level Specifications", Ph.D. Thesis, Stanford Artificial Intelligence Laboratory Memo AIM-282, Stanford Computer Science Report STAN-CS-76-560, Stanford University, Stanford, CA, July 1976.

60. Winey, III, C.M.
"Comptuer Simulated Visual and Tactile Feedback as an Aid to Manipulator and Vehicle Control", S.M. Thesis, MIT, Dept. of Mechanical Engineering, May, 1981.

61. Fyler, D.C.
"Computer Graphic Representation of Remote Environment Using Position Tactile Sensors", S.M. Thesis, MIT, Dept. of Mechanical Engineering, August 1981.

62. The touch panel display program was designed by Lisa Washington with support from the MIT Undergraduate Research Opportunities Program.

63. Donovan, J.J.
"Systems Programming," McGraw-Hill, New York, 1972.

64. The three axis force joystick was designed by Hans Griesser with support from the MIT Undergraduate Research Opportunities Program.

65. James, J.S.
"What is FORTH? A Tutorial Introduction", Forth Language Reprints, Forth Interest Group, PO Box 1105, San Carlos, CA., 1980.

66. Chapanis, A.
"Research Techniques in Human Engineering", Johns Hopkins Press, Baltimore, 1959.

67. Keppel, G.
"Design and Analysis: A Researcher's Handbook", Prentice-Hall Series in Experimental Psychology, 1973.

68. Schneiderman, B.
"Software Psychology - Human Factors in Computer and Information Systems", Winthrop Publishers, Cambridge, 1980.

69. Sime, M.E., Arblaster, A.T., Green, T.R.G.
"Reducing Programming Errors in Nested Contitionals by Prescribing a Writing Procedure", Int. J. Man-Machine Studies, Vol 9.1, pp 119-126, 1977.

70. Bertsche, W.R., Logan, K.P., Pesch, A.J., Wernli, R.L.
"Evaluation of the Design and Undersea Work Capability of the Work Systems Package", Naval Ocean Systems Center Technical Report 214, San Diego, April, 1978.

71. Hasegawa, T.
"An Interactive System for Modeling and Monitoring a Manipulation Environment", IEEE, J. Sys., Man, and Cyber., Vol. SMC-12, No. 3, May/June 1982, pp. 250-258.

72. Freedman, L.A., Crooks, W.H., Coan, P.P.
"TV Requirements for Manipulation in Space," Mechanism and Machine Theory, Vol. 12, 1977, pp. 425-438.

73. Heer, E. (editor)
"Remotely Manned Systems - Exploration and Operation in Space" Cal. Tech., 1973.

74. Maxwell, A.E.
"Multivariate Analysis in Behavioral Research", Chapman and Hall, London, 1977.

75. Busby, F.R. and Associates
"Underwater Inspection/Testing/Monitoring of Offshore Structures," US Dept. of Commerce, No. 7-35336, February, 1978.

76. Rock, I.
"An Introduction to Perception", Macmillan Publishing Co., New York, 1975.

DISTRIBUTION LIST

Capt. Paul R. Chatelier
Office of the Deputy Under
Secretary of Defense
OUSDRE (E&LS)
Pentagon, Room 3D129
Washington, DC 20301

Engineering Psychology Group
Office of Naval Research
Code 442EP
Arlington, VA 22217

Undersea Technology Programs
Code 220
Office of Naval Research
800 North Quincy Street
Arlington, VA 22217

Dr. Eugene Silva
Ocean Sciences, Code 421
Office of Naval Research
800 North Quincy Street
Arlington, VA 22217

Communication & Computer Technology
Programs
Code 270
Office of Naval Research
800 North Quincy Street
Arlington, VA 22217

Manpower, Personnel, and Training
Programs
Code 270
Office of Naval Research
800 North Quincy Street
Arlington, VA 22217

Information Sciences Division
Code 433
Office of Naval Research
800 North Quincy Street
Arlington, VA 22217

Special Assistant for Marine
Corps Matters
Code 100 M
Office of Naval Research
800 North Quincy Street
Arlington, VA 22217

Dr. J. Lester
ONR Detachment
495 Summer Street
Boston, MA 02210

Dr. Perry Alers
Marine Systems Branch, Code 5821
Naval Research Laboratory
Washington, D.C. 20375

Director
Naval Research Laboratory
Technical Information Division
Code 2627
Washington, D.C. 20375

Dr. J.E. Franklin
Navy Center for Applied Research in A.I.
Code 7510
Naval Research Laboratory
Washington, D.C. 20375

Office of the Chief of
Naval Operations
Deep Submergence Systems Division
Op-231
Washington, D.C. 20350

Dr. Robert G. Smith
Office of the Chief of Naval
Operations, OP 987H
Personnel Logistics Plans
Washington, D.C. 20350

Office of the Deputy Under
Secretary of Defense
OUSDRE (E&LS)
Pentagon, Room 3D129
Washington, D.C. 20301

CDR G. Worthington
Office of the Chief of Naval
Operations, OP 372G
Washington, D.C. 20350

Dr. Andreas B. Rechnitzer
Office of the Chief of Naval
Operations, OP 952F
Naval Oceanography Division
Washington, D.C. 20350

Technical Director
U.S. Army Human Engineering Labs
Aberdeen Proving Ground, MD 21005

U.S. Air Force Office of
Scientific Research
Life Sciences Directorate, NL
Bolling Air Force Base
Washington, DC 20332

Chief, Systems Engineering Branch
Human Engineering Division
USAF AMRL/HES
Wright-Patterson AFB, OH 45433

Director, Human Factors Wing
Defence & Civil Institute
of Environmental Medicine
Post Office Box 2000
Downsview, Ontario M3M 389
CANADA

Defense Technical Information Center
Cameron Station, Bldg. 5
Alexandria, VA 22314

Dr. Craig Fields
Director, System Sciences Office
Defense Advanced Research
Projects Agency
1400 Wilson Boulevard
Arlington, VA 22209

Dr. M. Montemerlo
Human Factors & Simulation
Technology, RTE-6
NASA HQS
Washington, DC 20546

Dr. J. Miller
Florida Institute of Oceanography
University of South Florida
St. Petersburg, FL 33701

Dr. Jesse Orlansky
Institute for Defense Analyses
1801 N. Beauregard Street
Alexandria, VA 22311

Human Factors Department
Code N215
Naval Training Equipment Center
Orlando, FL 32813

Mr. John Freund
NAVSEA 05R2
Naval Sea Systems Command
Washington, DC 20362

Dr. Gary Poock
Operations Research Department
Naval Postgraduate School
Monterey, CA 93940

Mr. H. Talkington
Ocean Engineering Department
Naval Ocean Systems Center
San Diego, CA 92152

Mr. Paul Heckman
Naval Ocean Systems Center
San Diego, CA 92152

Mr. Warren Lewis
Human Engineering Branch
Code 8231
Naval Ocean Systems Center
San Diego, CA 92152

Dr. Ross L. Pepper
Naval Ocean SYstems Center
Hawaii Laboratory
P.O. Box 997
Kailua, HI 96734

Dr. A.L. Slafkosky
Scientific Advisor
Commandant of the Marine Corps
Code RD-1
Washington, DC 20380

Human Factors Technology Administrator
Office of Naval Technology
Code MAT 0722
800 N. Quincy Street
Arlington, VA 22217

Mr.Phillip Andrews
Naval Sea Systems Command
NAVSEA 0341
Washington, DC 20362

Mr. Milon Essoglou
Naval Facilities Engineering Command
R&D Plans and Programs
Code 03T
Hoffman Building II
Alexandria, VA 22332

CDR Robert Biersner
Naval Medical R&D Command
Code 44
Naval Medical Center
Bethesda, MD 20014

Dr. Arthur Bachrach
Behavioral Sciences Department
Naval Medical Research Institute
Bethesda, MD 20014

CDR Thomas Berghage
Naval Health Research Center
San Diego, CA 92152

Dr. George Moeller
Human Factors Engineering branch
Submarine Medical Research Lab
Naval Submarine Base
Groton, CT 06340

Dr. Robert Blanchard
Navy Personnel Research and
Development Center
Command and Support Systems
San Diego, CA 92152

Mr. John Quirk
Naval Coastal Systems Laboratory
Code 712
Panama City, FL 32401

Dr. Edgar M. Johnson
Technical Director
U.S. Army Research Institute
5001 Eisenhower Avenue
Alexander, VA 22333

Dr. Harry Snyder
Department of Industrial Engineering
Virginia Polytechnic Institute and
State University
Blacksburg, VA 24061

Dr. Robert T. Hennessy
NAS - National Research Council
2101 Constitution Ave., N.W.
Washington, DC 20418

Dr. Amos Freedy
Perceptronics
6271 Variel Avenue
Woodland Hills, CA 91364

Dr. Edward R. Jones
Chief, Human Factors Engineering
McDonnell-Douglas Astronautics Co.
St. Louis Division
Box 516
St. Louis, MO 63166

Dr. Richard W. Pew
Information Sciences Division
Bolt Beranek & Newman, Inc.
50 Moulton St.
Cambridge, MA 02138

Dr. A. K. Bejczy
Jet Propulsion Laboratory
California Institute of Technology
Pasadena, CA 91125

Dr. Robert D. Ballard
Department of Ocean Engineering
Woods Hole Oceanographic Institution
Woods Hole, MA 02543